Smart Object Development Kit
GEN3

User Guide

Version 3.4

www.centerotech.com

www.centerotech.com

Table of Contents

1 Introduction 6

1.1 Document purpose 6

1.2 Audience 6

1.3 Abbreviations and acronyms 6

2 Development Kit Overview 7

2.1 About the Smart Object Development Kit 7

2.2 Smart Object platform architecture 9

2.3 Standards implemented in Smart Object platform 10

2.4 Platform topology 11

2.5 Summary of the Smart Object Development Kit functional features 12

2.5.1 The Smart Object 12

2.5.2 The Smart Object endpoint 15

2.5.3 The Versa Router™ 1000 Quark 15

2.5.4 The Network and Application Monitoring Tool 17

2.6 Development Kit contents 18

3 Getting Started 19

3.1 Overview 19

3.2 Powering up the system components 21

3.2.1 Powering up the VersaRouter™ 1000 21

3.2.2 Powering up the Smart Objects 23

3.2.3 Joining Smart Objects to the Versa Router™ 1000 Quark 24

3.3 Connecting to the Monitoring Tool (NAMT) 25

3.3.1 Configuring the PC static IP address to access the Quark 25

3.3.2 Optional: Ensure the Quark is accessible from the NAMT PC 27

3.3.3 Installing the NAMT 28

3.3.4 Starting up the NAMT 29

4 Development Kit Components - Hardware 31

4.1.1 Smart Object Endpoint 31
4.1.2 Versa Router™ 1000 Quark hardware description 42
5 Development Kit Components - Software 47
5.1.1 The Application Processor Interface (API) 47
5.1.2 The Versa Router™ 1000 Quark Software 49
5.1.3 The NAMT 91
5.2 Set-up Quark / Cloud-Based NOC communication ..... **Error! Bookmark not defined.**
5.2.1 Add Quark in the Cloud-Based NOC Whitelist ..... **Error! Bookmark not defined.**
5.2.2 Set-up Quark and NMS communication using VPN [RECOMMENDED] **Error! Bookmark not defined.**
5.2.3 [Alternate] Set-up Quark-NMS plain-text communication (Use for lab tests only) **Error! Bookmark not defined.**
6 Upgrading the Development Kit 126
6.1 Overview 126
6.2 Upgrading the Smart Object stack firmware using serial line 126
6.3 Upgrading the Smart Object stack firmware OTA using NAMT 129
6.4 Upgrading the Smart Object stack firmware OTA using Cloud-based NOC website 129
6.5 Upgrading the Quark Transceiver firmware using Quark web interface 129
6.6 Upgrading the Quark Transceiver firmware using Quark /admin/ interface 129
6.7 Upgrading the Versa Router™ 1000 Quark software using Quark web interface 129
6.8 Upgrading the Versa Router™ 1000 Quark software using Quark /admin/ interface 129
6.9 Upgrading the Versa Router™ 1000 Quark software using Cloud-based NOC website 129
6.10 Upgrading the Versa Router™ 1000 Quark website using Quark /admin/ interface 130
6.11 Upgrading the NAMT 130
7 Configuring the Smart Objects Development Kit 131
7.1 Overview 131
7.2 Accessing the configuration interface on the Smart Object 131
7.3 Accessing the configuration interface on the Quark 136
7.4 Enable/Disable DTLS, PANA and Low Power 137
7.4.1 Overview 137
7.4.2 Enable/Disable low power on Smart Objects – VN400 flavor only 137
7.4.3 Enable/Disable DTLS and PANA on the Smart Objects 138
7.4.4 Enable/disable DTLS on the Quark 139
7.4.5 The Smart Objects certificates 141

7.5 Change the DTLS Certificate / pre-shared Key 144
7.5.1 Overview 144
7.5.2 Change the DTLS Key/certificate on the Smart Object 144
7.5.3 Change the DTLS Key/Certificate on the Quark 145
7.6 Change the PANA Certificate / pre-shared Join Key 147
7.6.1 Choose the PANA type on the Smart Object 147
7.6.2 Change the PANA Certificate 147
7.6.3 Change the pre-shared Join Key 149
7.7 Change the Network ID 151
7.7.1 Overview 151
7.7.2 Change the Net ID on the Smart Object 151
7.7.3 Change the Vendor Network ID on the Quark 152
7.8 Set-up Quark / Cloud-Based NOC communication 153
7.8.1 Add Quark in the Cloud-Based NOC Whitelist 153
7.8.2 Set-up Quark and NMS communication using VPN [RECOMMENDED] 155
7.8.3 [Alternate] Set-up Quark-NMS plain-text communication (Use for lab tests only) 161
8 Use Cases 166
8.1 Setting the PC-to-device communication 166
8.1.1 PC-to-device IPv6 communication for Windows user, VPN based 166
8.1.2 PC-to-device IPv6 communication for Windows user, non-VPN based 172
8.1.3 PC-to-device IPv6 communication for Linux user, VPN based 174
8.1.4 PC-to-device IPv6 communication for Linux user, non-VPN based 179
8.2 Evaluating the on-board application COAP mapped parameters 181
8.3 Application Processor Mapped COAP parameters 183
8.3.1 Overview 183
8.3.2 Resource Discovery 183
8.3.3 Read Resource 184
8.3.4 Observe Resource 185
8.3.5 Retrieve Device History 186
8.3.6 Get network topology/device-list 187
9 Troubleshooting 188
Appendix A: List of Standards Supported in the Smart Object Platform 189
Appendix B: FCC Related Statements 190

Appendix C: Industry Canada Related Statements 191
Appendix D: European (ETSI) Related Statements 194

1 Introduction

1.1 Document purpose

This user guide describes the content and use of the Gen3 M1 Smart Object development kit, including hardware and software installation, configuration, and use of the Network and Application Monitoring Tool.

1.2 Audience

This document is intended for the users of the Gen3 M1 Smart Object Integration Kit.

1.3 Abbreviations and acronyms

Versa Router™ 1000 Quark	The Versa Router™ 1000 Quark is comprised of the Phytec-designed and built phyCORE-AM335x, with “Smart Object” attached for the Transceiver.
VR1000	Short name for Versa Router™ 1000 Quark
Quark	Short name for Versa Router™ 1000 Quark
Edge Router	Short name for Versa Router™ 1000 Quark
phyCORE-AM335x	Phytec-designed and built board based on AM335x processor
Network and Application Monitoring Tool	Windows-based monitoring and management tool supplied by Nivis that will to allow the end user to evaluate the performance of the Smart Object network
Power Pack	Nivis-designed board that will supply either DC or battery power to the “Tower Plug-In Module” and the Nivis “Smart Object” radio module
Smart Object	Nivis-built, manufactured, and sold Kinetis + Navajo Radio Module
Smart Object API Documentation	Document supplied by Nivis that specifies the API through which an application processor can communicate with the Smart Object in order to send and receive application-and network-related status data.
Smart Object Development Kit	Consists of 1 Quark and 2 battery-powered and 2 line-powered “Smart Object End Nodes.” The “Network and Application Monitoring Tool” is a Windows-based application supplied on a USB stick. Also included in the Kit are the “Quick Start Guide,” “User Guide,” “Smart Object API Documentation,” and “Smart Object Datasheet.”
Smart Object End Node	Combination of the “Smart Object” radio module, plus the “Quark,” plus the “Power Pack.” The end node can either be line-powered by DC input or battery-powered by 2 AA batteries.

2 Development Kit Overview

2.1 About the Smart Object Development Kit

The Smart Object Development Kit (SODK) is a user-friendly and versatile development kit that serves two purposes.

The main purpose of the kit is to allow interested parties to integrate Nivis' third generation wireless Smart Object technology into their products quickly and easily. The Smart Objects are running on a Freescale-based state-of-the-art sub-Gigahertz hardware platform; the Quark Router is running on a Phytec AM335x board. The user can access the Smart Object stack via a UART or SPI serial communication port, allowing the user to connect an application processor to the stack modem. With minimal firmware development effort and following a simple API, the user can swiftly build a wireless product that is an integral part of the Internet of Things.

The secondary purpose of the SODK is to allow users to evaluate the performance of:

- The Smart Object hardware platform
- The Quark Router hardware platform
- Nivis' third generation Smart Object wireless networking platform

The Nivis Smart Object platform is a wireless communication platform based on IEEE and IETF standards for low-power, short-range, resource-constrained wireless devices. It enables building wireless Smart Objects through which the Internet is extended to small, embedded objects that monitor and control various parameters in our surroundings.

The communication platform was architected and developed to support the following key design goals.

Design Goal	Market Benefits
High scalability	Support for 500 smart objects per Quark minimizes the cost of supporting infrastructure and allows for wide geographic coverage.
Standards-based platform	An entirely standards-based solution ensures cross-vendor interoperability and hence offers consumer advantages achieved by encouraging competition in the market. It also encourages the establishment of an ecosystem of interoperable Smart Objects while ensuring the transparency of the solution. Finally, standards compliance facilitates regulatory compliance.
Reliable delivery of data	The Nivis Smart Object platform targets communications over short-range wireless links in which reliable delivery of data is paramount. Data transmission reliability is ensured through various mechanisms such as: • Path diversity (mesh) achieved through compliance with the IETF ROLL-defined RPL routing protocol • Time division multiplexed bandwidth allocation • Frequency diversity (hopping) • ARQ (automatic repeat request) • Hop-to-hop acknowledged delivery • End-to-end acknowledged delivery
Smart Objects incorporate 6loWPAN/IPv6 connectivity	IP connectivity to the Smart Object ensures that the device becomes an integral part of the Internet through IP addressability and connectivity. It also allows end users to leverage IPv6 network management and diagnostic tools. The 6LoWPAN header compression scheme allows for transmission of shorter packets, which in turn maximizes the wireless bandwidth available and increases the battery life of battery-powered Smart Objects.
Smart Objects incorporate COAP/REST-based web services	Extending support for web services to the Smart Object allows the device to participate directly in the World Wide Web.
Application agnostic	Offering an application-agnostic communication platform ensures that the final solution is similar in charter to the Internet and the Internet of Things. A common infrastructure supports a wide variety of applications, since the communication platform is tunable to optimized application requirements.
Class-leading security	Ensures authenticity, confidentiality, and integrity of the data delivered through link layer security mechanisms using PANA for key exchange and DTLS end-to-end security
Power diverse solution	Support for both line-powered and battery-powered devices.

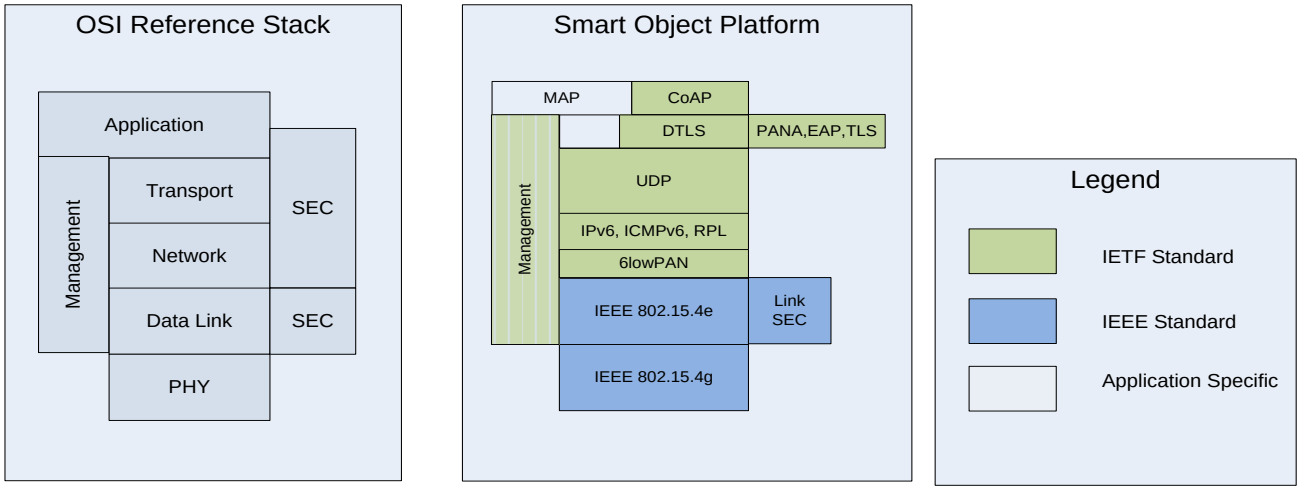
2.2 Smart Object platform architecture

The Nivis Smart Object platform is an application-agnostic wireless monitoring and control platform. The entire communication stack construct is standards-based.

As suggested previously, an entirely standards-based solution ensures:

- Cross-vendor interoperability
- Fulfillment of end-user requirements
- Encouragement of competition in the market
- Consumer cost advantages
- Establishment of an ecosystem of interoperable devices
- Transparency of the solution
- Regulatory compliance

The standards-based Smart Object platform, side-by-side with the OSI Reference Platform, is illustrated in the figure below:



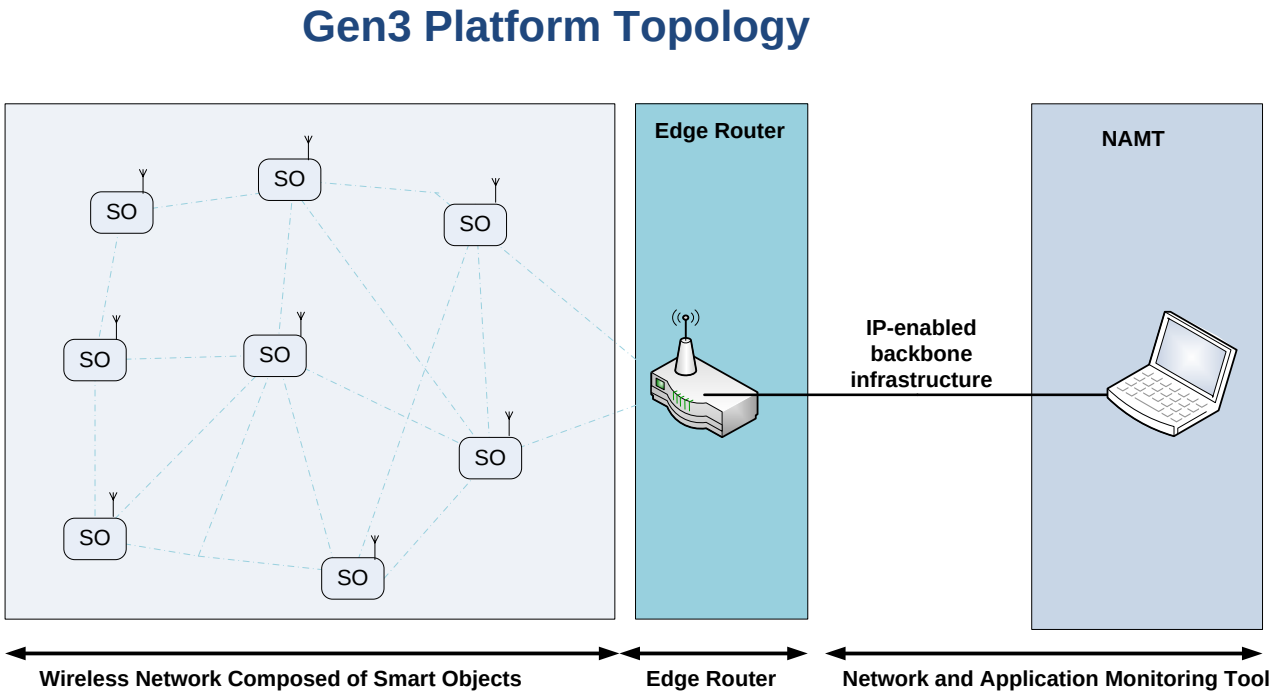
2.3 Standards implemented in Smart Object platform

The standards implemented into the Smart Object platform include the following:

Standardization Body	Standard Designator	Revision	Title
IEEE	802.15.4g-2012	Final	Low-Rate Wireless Personal Area Networks (WPANs)Amendment 4: Physical Layer Specifications for Low Data Rate Wireless Smart Metering Utility Networks
IEEE	802.15.4e-2012	Final	Wireless Medium Access Control(MAC) and Physical Layer (PHY)Specifications for Low-Rate Wireless Personal Area Networks (WPANs)
IETF	RFC 2460	Final	Internet Protocol, Version 6 (IPv6)Specification
IETF	RFC4443	Final	Internet Control Message Protocol for the IPv6 Specification
IETF	RFC 4944	Final	Transmission of IPv6 Packets over IEEE 802.15.4 Networks
IETF	RFC 6282	Final	Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks
IETF	RFC 6775	Final	Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)
IETF	RFC 6550	Final	RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks
IETF	RFC6202	Final	Trickle Algorithm
IETF	RFC 6552	Final	Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)
IETF	RFC 6551	Final	Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks
IETF	RFC768	Final	User Datagram Protocol (UDP)
IETF	ID.draft-ietf-core-coap	Rev 16	Constrained Application Protocol (CoAP)
IETF	RFC 6690	Final	Constrained RESTful Environments (CoRE) Link Format
IETF	ID.draft-ietf-core-observe	Rev 8	Observing Resources in CoAP
IETF	RFC 6347	Final	Datagram Transport Layer Security Version 1.2
IETF	RFC 5191	Final	Protocol for Carrying Authentication for Network Access (PANA)
IETF	RFC 3748	Final	Extensible Authentication Protocol (EAP)
IETF	RFC 5216	Final	EAP-TLS Authentication Protocol
IETF	RFC 5246	Final	Transport Layer Security (TLS) Protocol Version 1.2
W3C	W3C Recommendation 10 March 2011	1.0	Efficient XML Interchange (EXI)

2.4 Platform topology

The platform topology is illustrated in the image below.



2.5 Summary of the Smart Object Development Kit functional features

2.5.1 The Smart Object

The list below presents the main features and mechanisms present in the wireless Smart Object modem.

NOTE 1 For a more detailed description of the Smart Object hardware platform, please consult Development Kit Components - Hardware. **NOTE 2** For detailed hardware specifications and parameters, please consult the VN400 – Smart Object datasheet provided with the kit.

Layer	Functional Feature	Notes
PHY	Wireless communication is IEEE 802.15.4g-compliant and utilizes the sub-Gigahertz frequency spectrum as mandated by the standard. It utilizes the MR-FSK physical layer.	US and Canada: 902 – 928 MHz Japan: 922.3 – 928.1 MHz Europe: 863 – 870 MHz
PHY	Wireless communication is compliant with the header structures, timings, and PIBs present in the IEEE 802.15.4g standard.	
PHY	The Smart Object supports both the Common Signaling Mode (CSM) and mandatory higher bitrates as defined in the IEEE 802.15.4g standard. It also supports dynamically switching between CSM and the higher bitrate.	US and Canada: CSM and 200 kbps Japan: CSM and 100 kbps Europe: CSM and 100 kbps
PHY	The Smart Object is compliant with applicable regulatory requirements in the United States, Canada, Europe, and Japan.	US: FCC (Title 47, part 15) Canada: IC RSS-210 Japan: ARIB T108 Europe: ERC 70-03/ETSI EN300-220
PHY	Supports payloads as large as 2048 bytes.	
MAC	The MAC layer is IEEE 802.15.4e – TSCH compliant. It utilizes the MIBs, information elements, and header structures present in the IEEE 802.15.4e amendment.	
MAC	The Smart Object supports frequency diversity through channel hopping in order to: 1. Minimize susceptibility to interference from other devices that operate in the same frequency spectrum 2. Maximize the number of coexisting Smart Object networks that operate in the same vicinity	US and Canada: 64 channels Japan: 28 channels Europe: 64 channels

Layer	Functional Feature	Notes
MAC	The Smart Object utilizes TDMA in order to utilize the frequency spectrum most efficiently and maximize the aggregate bandwidth of the WLAN composed of Smart Objects.	
NWK	The Smart Object utilizes IPv6 packets and addressability and the IETF-defined 6LoWPAN compression for constrained wireless Smart Objects.	
NWK	Supports IPv6 address auto-configuration.	
NWK	Supports the widely used ping utility.	
NWK	Supports path diversity by implementing the RPL protocol, resulting in maximized reliability of data transmissions.	The WLAN formed of Smart Objects is a multi-hop network that can be as deep as 25 hops.
NWK	Inbound routing is based on RPL-defined graphs (DODAG) that are formed utilizing ICMPv6-based discovery. Supports ROLL-defined Objective Function 0. Also supports concurrent operation in multiple DODAGs.	
NWK	Outbound routing is based on both RPL-defined storing mode (ensuring outbound path redundancy) and source routing.	
TL	Communication at the transport layer utilizes widely-used UDP datagram's.	
AL	Supports COAP/HTTP based web services including the COAP-defined observable mechanism that allows constrained wireless Smart Objects periodically to publish parameters of interest without incurring the penalties associated with the HTTP-based request/response data model.	
AL	All application layer parameters are modeled as COAP web resources and are directly addressable from any browser.	
AL	Communicates with an external application processor based on a simple API over UART or SPI.	
AL	Allows an external application processor to map up to four parameters into COAP-modeled web resources that can be monitored via the NAMT.	

Layer	Functional Feature	Notes
AL	Monitors and periodically publishes the following application layer COAP modeled resources: 1. Digital status of a button 2. Analog status of a potentiometer 3. Power status a. Line-powered b. Battery-powered – battery level and critical alert	
AL	Application layer payloads are encoded utilizing EXI, which is a compressed form of XML.	
MGMT	Management structures are modeled as COAP resources that are accessible from any web browser.	
MGMT	The Smart Object periodically publishes the following statistics (all modeled as COAP web resources): 1. Channel statistics – PER, which is indicative of the quality of communication on each channel 2. Neighbor statistics – PER, which is indicative of the quality of the communication link with individual neighbors	
MGMT	The Smart Object communication stack is remotely upgradeable over-the-air.	
SEC	Utilizes link layer hop-by-hop IEEE 802.15.4-defined security, including authentication and AES-128 encryption.	

2.5.2 The Smart Object endpoint

The Smart Object endpoint consists of the Smart Object modem that is hosted by a Power Pack board. The functional features of the Power Pack board are captured in the table below.

NOTE For a more detailed description of the Power Pack board hardware, please consult Development Kit Components - Hardware.

Functional Feature	Notes
Can be line-powered using a mini-USB power supply (provided with the kit) or battery-powered. Battery-powered Smart Object endpoints allow for increased mobility, which is useful when evaluating the RF behavior of the Smart Object.	The Smart Object automatically and dynamically detects the power mode at start-up.
Battery power is continuously monitored and reported. The results can be visualized in the NAMT. Power status is signaled through an LED.	Also includes a low battery alert.
On-board toggle switch allows the user to select between UART-and SPI-based communication with an external application processor.	
On-board status button and LED indicates the network status of the Smart Object (in discovery, joined, etc.).	
On-board potentiometer allows the user to highlight changes of an analog parameter.	
Hosts a push-button that allows the end user to reset the Smart Object modem.	
On-board TWRPI expansion board allows the connection of various TWRPI modules designed and marketed by Freescale.	

2.5.3 The Versa Router™ 1000 Quark

The Versa Router™ 1000 Quark is the network entity responsible for arbitrating and managing the WLAN formed of Smart Objects. It also mitigates between the WLAN and entities present on the backbone infrastructure (such as the NAMT). The functional features of the Quark are described in the table below.

NOTE For a more detailed description of the Versa Router™ 1000 Quark hardware platform, please consult Development Kit Components - Hardware.

Functional Feature	Notes
Supports connectivity between the Smart Object WLAN and entities residing on the Internet.	A good example is the PC that hosts the Network and Application Monitoring Tool (NAMT).
Provides central arbitration for the WLAN formed of Smart Objects by acting as a MAC WLAN coordinator.	This is accomplished through the distribution of an 802.15.4e-compliant network maintenance slotframe.
Collects network and communication diagnostics sent by Smart Objects, such as: 1. Channel statistics 2. Neighbor-related statistics 3. Routing (RPL) -related information such as the topology of the network	Network- and communication-related statistics and parameters are displayed in the NAMT.
Acts as the link layer security manager and the termination of hop-to-hop security.	
Acts as an extraction point for application-related as well as management parameters. Parameters are extracted utilizing HTTP requests and methods via a COAP/HTTP proxy.	
Hosts an EXI/XML translator, allowing external entities to extract XML-encoded application and management payloads.	
Host a web server for router administration: management of network profiles, whitelist, configuration parameters, and network configuration of the Router	
Can store application layer payloads for up to 24 hours for up to 500 Smart Objects with a reporting interval of 1 minute. Historical data can be retrieved from HTTP/COAP Proxy using XML over HTTP requests.	
Allows management for application-related parameters: • Allows extraction of the most recent as well as historical COAP resources (up to 24 hours old), translating from EXI to XML if necessary • Allows observation of (subscription to) COAP-observable resources on the devices • Allows getting and setting of the reporting interval for observable resources	

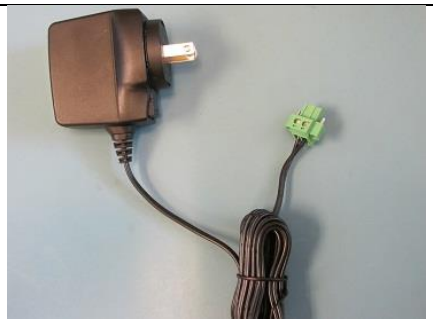
2.5.4 The Network and Application Monitoring Tool

The NAMT (Network and Application Monitoring Tool) is a stand-alone Windows-based application that ships with each development kit. The NAMT is the configuration, visualization, and diagnostic tool for the kit, with features as described in the table below.

Functional Feature	**Notes**
Visualization of the mechanisms associated with time-synchronized channel hopping operation	
Time-slotted (TDMA) communication schedule of the Smart Objects	
Channel quality map that shows packet success rates per channel	
Visualization of the remaining battery life of the Smart Objects	
Visualization of the topology of the WLAN composed of Smart Objects (RPL tree structure)	
Visualization of various management parameters of the Smart Objects	IPv6 address, EUI-64 address, Join status
Visualization of various communication diagnostics of the Smart Objects	Packet success rate per channel; packet success rates associated with communication with particular neighbors
Send on-demand data read commands	
Firmware upgrade of the Smart Objects, in both P2P and Broadcast mode; Broadcast can be sent with or without a filter (with filter, it is propagated to the whole network, but applied on a subset of devices only).	With coexistence of multiple HW/SW versions in the same network, Filtering feature allows targeting the upgrade to a specific subset of devices, characterized by HW type or SW version.
Firmware upgrade of the transceiver	
Visualization of resources	Non-observable and non-observed resources are only read once to reduce RF traffic.
Visualization of observed resources history	History is read from Quark data history storage, constructed based on published values from the devices
Subscribe/unsubscribe to observable resources	
Setting the publish interval of the subscribed observable resources	

2.6 Development Kit contents

The Smart Object Development Kit includes the components and associated quantities listed below.

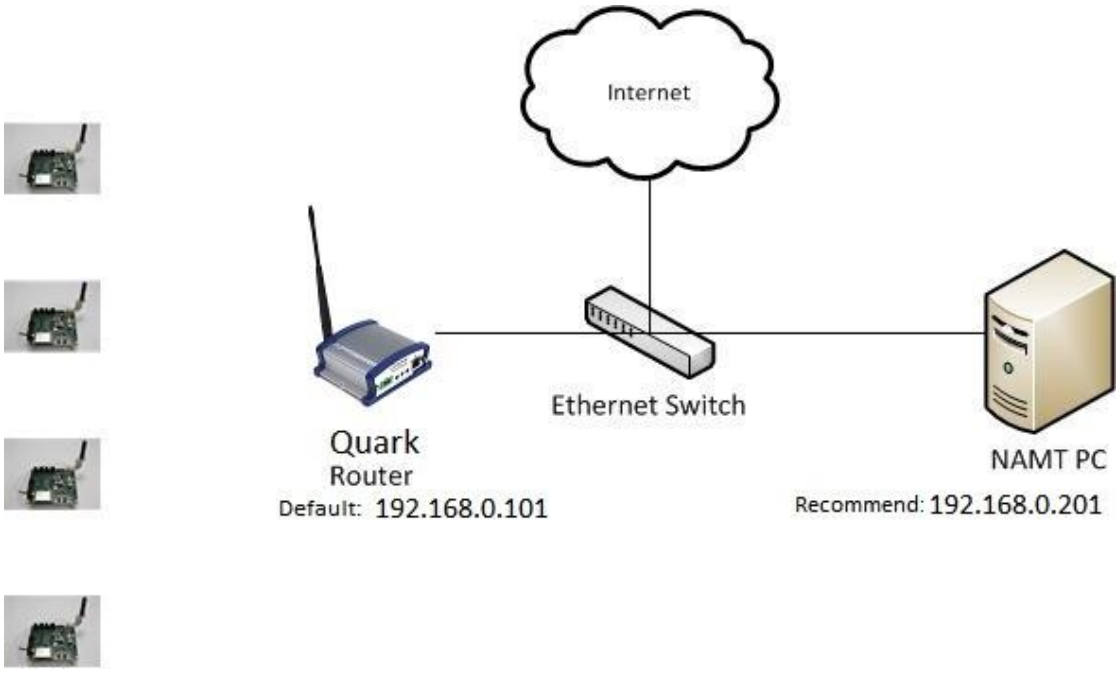
Component	Quantity	Picture
Smart Objects Radio Module	4	(several pictures are included later in this document)
Versa Router™ 1000 Quark	1	
Antenna (PN S467AH-915S)*	1	The connected antenna is shown with the Versa Router™ 1000 Quark above.
Cables (USB)		The connected cable is shown with the Freescale tower setup.
Power Adapter	1	

3 Getting Started

3.1 Overview

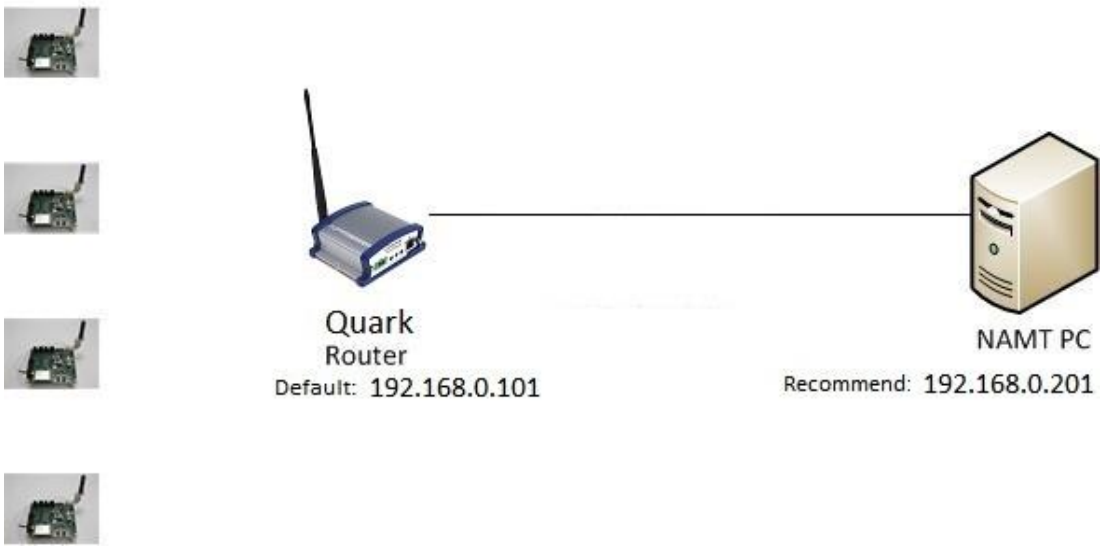
The Versa Router™ 1000 Quark and the NAMT PC can connect either directly through a cross-over cable (not recommended, as this has drawbacks, most notably slow boot-up and incorrect data timestamp), or using an Ethernet switch (recommended).

The recommended setup uses a switch, which also connects the kit to the Internet as illustrated in the figure below.



See the Quick Start Guide for the default Versa Router™ 1000 Quark IPv4 and other necessary network configuration information.

The alternate setup does not require a switch. This setup is possible but **not recommended**, as in this setup the time cannot be synchronized with Internet time servers; therefore, the communication timestamps will be incorrect.



Steps to a functional kit:

1. Assemble the setup components: Versa Router™ 1000 Quark, Smart Objects
2. Power up the setup components: Versa Router™ 1000 Quark, Smart Objects
3. Configure the Versa Router™ 1000 Quark
4. Optional – Install the NAMT (you can skip this step if you only extract the data via XML/HTTP or EXI/COAP interfaces and do not use the NAMT interface to monitor the network, configure observables, and extract data)
5. Optional – Connect the NAMT with the Versa Router™ 1000 Quark (default IP: 192.168.0.101)

Step-by-step description of how to connect the components is described below.

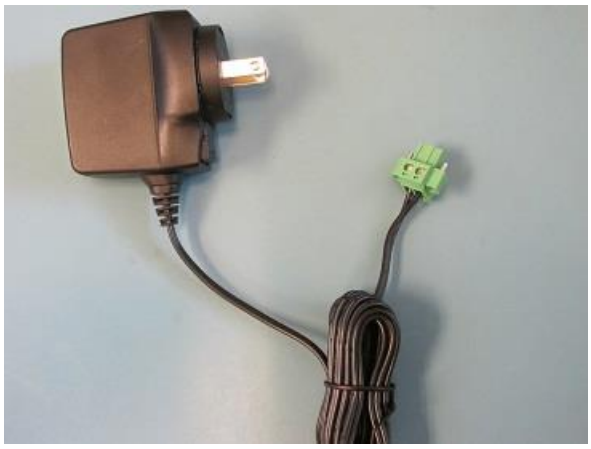
3.2 Powering up the system components

3.2.1 Powering up the VersaRouter™ 1000

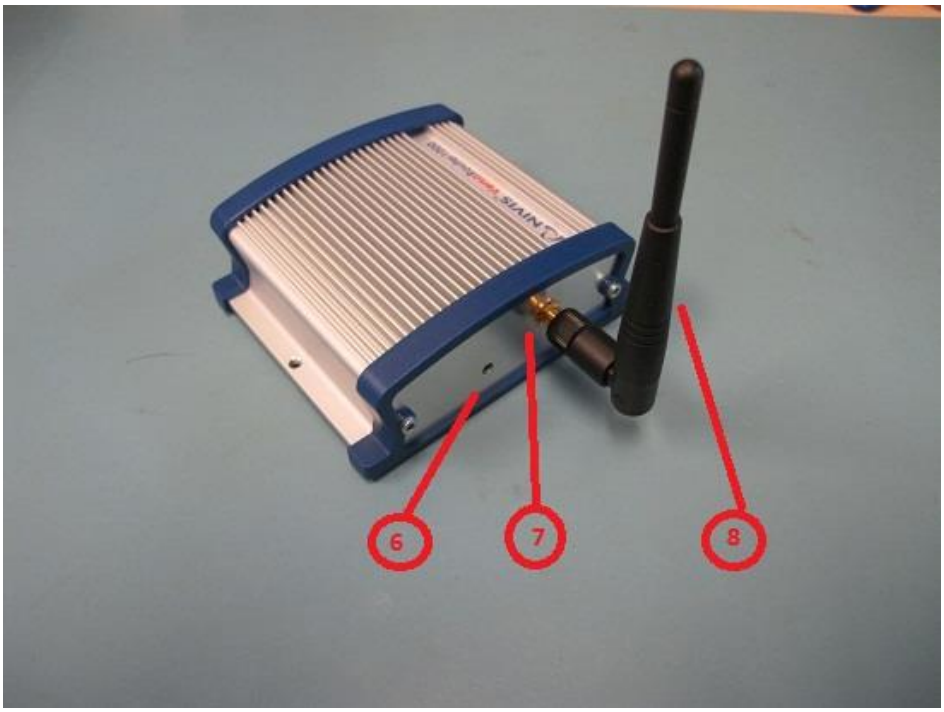
1. Attach one end of an Ethernet Cable to the Ethernet port (5). Insert the other end of an Ethernet cable to network equipment (Ethernet switch or hub).

   OPTIONAL: An external Ethernet router is not absolutely necessary but is highly recommended; instead of connecting to the external router, the Ethernet cable (cross-over) can be connected directly to the PC. One of the drawbacks is that in this case, the Versa Router™ 1000 Quark boot time will increase.

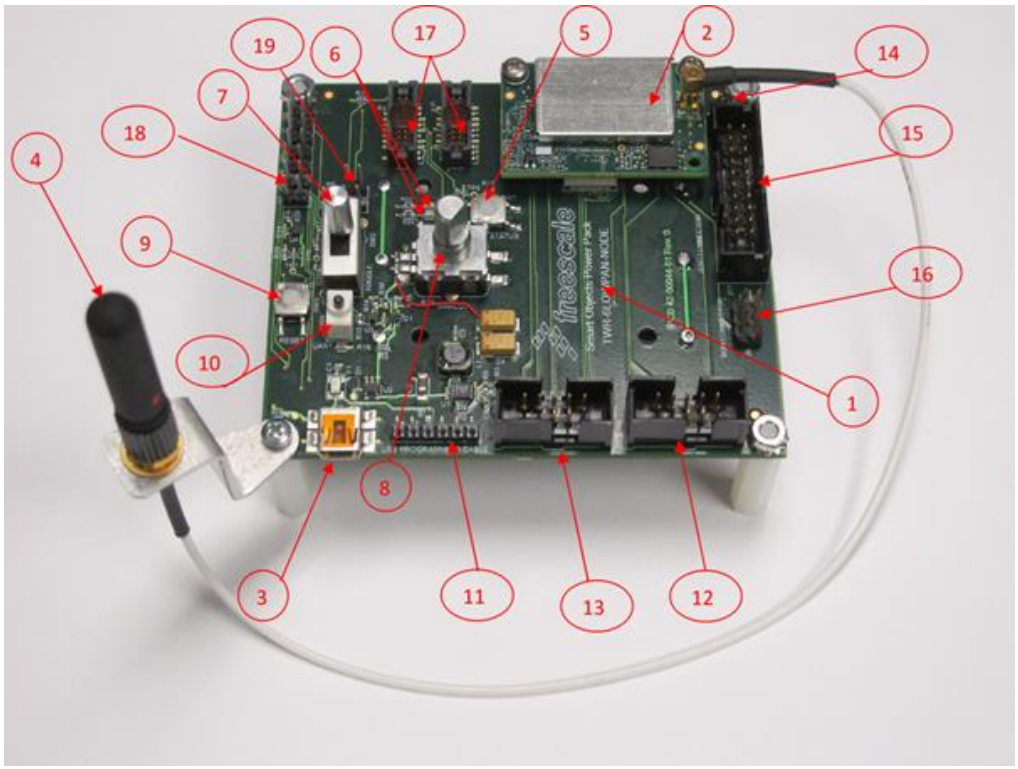
2. Attach the AC/DC Power Adapter barrel to the power connector of the Versa Router™ 1000 Quark (1). This is located on the left side.



3. Connect the Power Adapter to the AC socket or cord and confirm that the LED “POWER” on Quark (2) lights up.
4. After few seconds, observe the LEDs installed on the Ethernet port. One LED will be in a solid ON state, which indicates a link to the Ethernet network. The second LED will blink, indicating network activity.
5. Connect the antenna (8) to the antenna connector (7).



3.2.2 Powering up the Smart Objects



1. For all battery-powered Smart Objects devices, insert the AA batteries into battery holder BH1 located on the bottom side of the Power Pack. Install batteries with respect to the polarity marked inside the holder.
2. For all line-powered Smart Objects devices, attach the AC/DC Power Supply's mini-USB connector to the mating mini-USB socket located on the Power Pack (3). Connect the AC/DC Power Supply to the AC power line.

NOTE Do NOT install batteries on line-powered Smart Objects.

3. Install power-on jumper (14) (J18 installed, is ON).
4. Observe the green-colored "JOIN STATUS LED" (6); it should start blinking. This LED will blink for approximately ten seconds, then will turn off to conserve power. In order to check the status, the operator needs to push the momentary STATUS button (5); the LED will light up to indicate the JOIN STATUS of the Smart Object. A blinking LED indicates a "Not Joined" status.
5. Re-check the JOIN STATUS by pressing the STATUS button (5) and observing the LED behavior until the Green LED remains in a Solid "ON" state. This indicates a "Joined" status for the Smart Object.
6. At this point, the Smart Object is Joined and sensors can be used at will while observing the response of sensor actions in the NAMT (after connecting the NAMT to the Versa Router™ 1000 Quark, see The NAMT).

3.2.3 Joining Smart Objects to the Versa Router™ 1000 Quark

For Smart Objects to join a Versa Router™ 1000 "Quark," the **Vendor ID on the Versa Router™ 1000 Quark must match the Net ID on the Smart Objects**. By using the defaults, the Smart Objects and the Versa Router™ 1000 Quark settings are matched.

However, in the case of two networks coexisting in the same area, the user must change both settings on at least one of the networks. To enable coexistence, the PAN ID and the hopping sequence of the two controlling Quarks must be different.

We recommend changing the FAR ID of one of the routers and then using "Based on FAR ID" for the PAN ID, Vendor Network ID, and Hopping Sequence ID. For additional details, see Configuring the Versa Router™ 1000 Quark Identifiers.

- See documentation provided with the Smart Objects for instructions how to change the Net ID on the Smart Objects.
- See SMO_Kit_Configuration.docx for instructions how to change the Net ID on the Smart Objects.

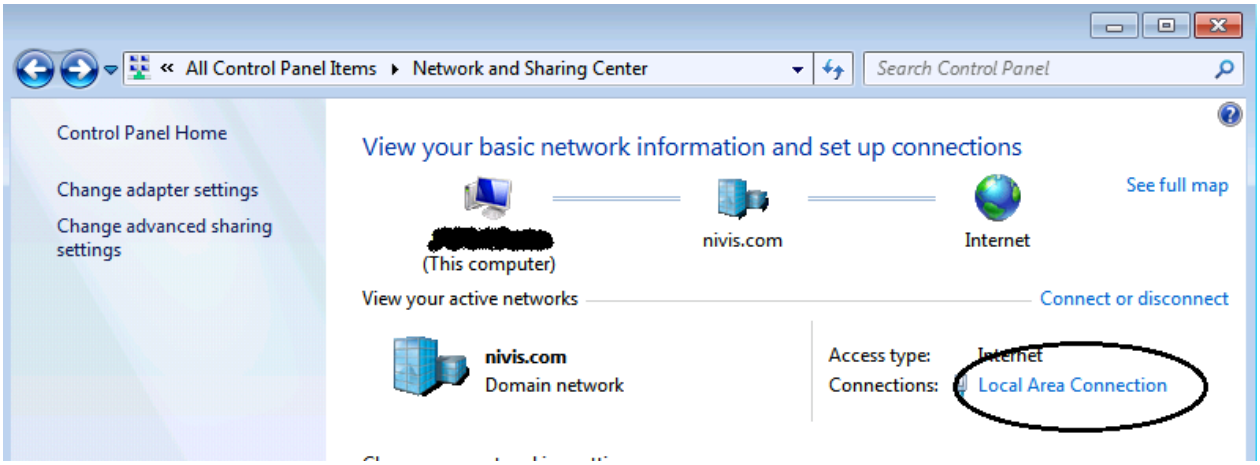
Visit the Nivis website at www.nivis.com to purchase additional Smart Objects.

3.3 Connecting to the Monitoring Tool (NAMT)

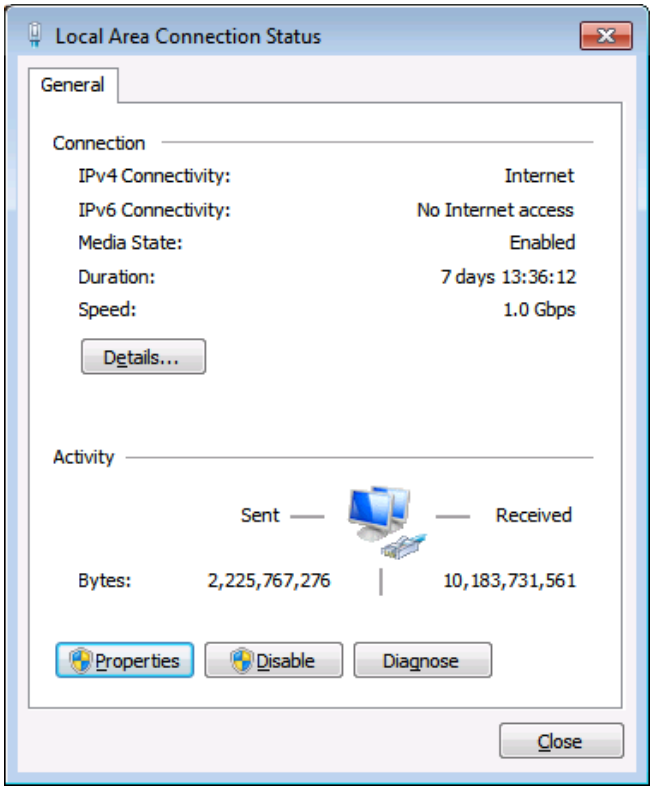
3.3.1 Configuring the PC static IP address to access the Quark

Step-by-step instructions for Windows 7:

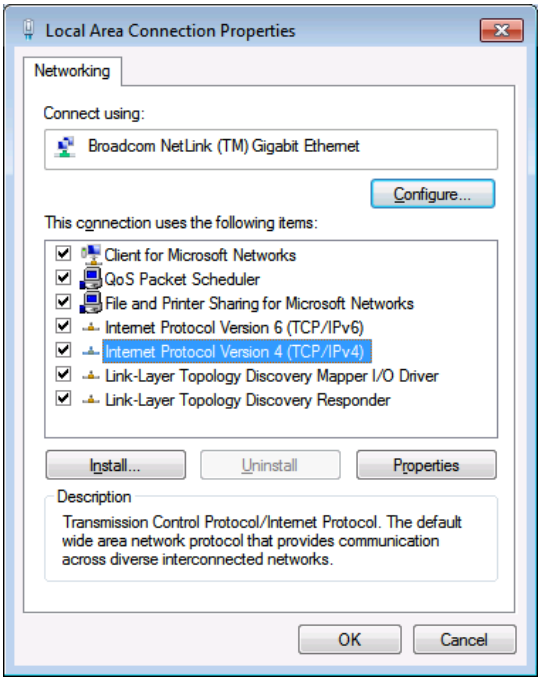
1. Open **Control Panel→Network and Sharing Center** and click on **Local Area Connection**.



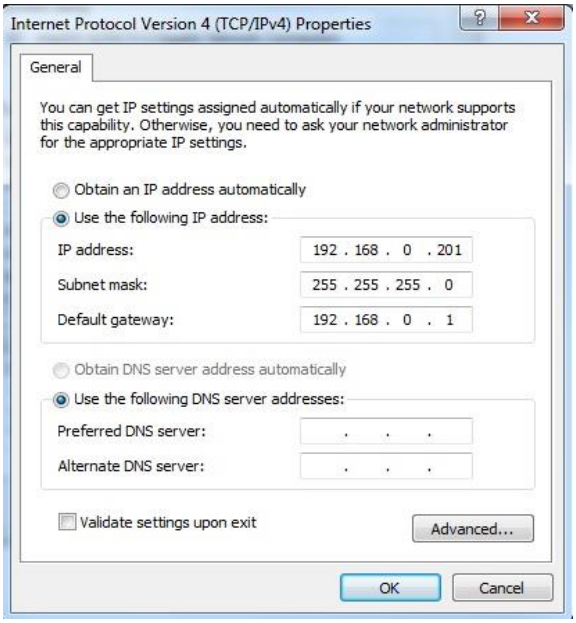
2. Click on **Properties**.



3. Click on **Internet Protocol Version 4 (TCP IPv4)** and then click on **Properties**.



4. Enter **IPv4: 192.168.0.201**, **Subnet Mask: 255.255.255.0**, **Default Gateway 192.168.0.1**. (**NOTE** Any other available IPv4 EXCEPT 192.168.0.101/192.168.0.1 can be used as PC IPv4).

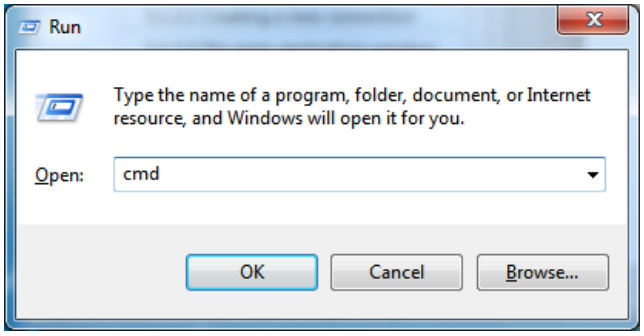

Internet Protocol Version 4 (TCP/IPv4) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 201
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 1
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server:
Alternate DNS server:
Validate settings upon exit
Advanced...
OK
Cancel

5. Check **Validate settings upon exit**.
6. Click **OK**.

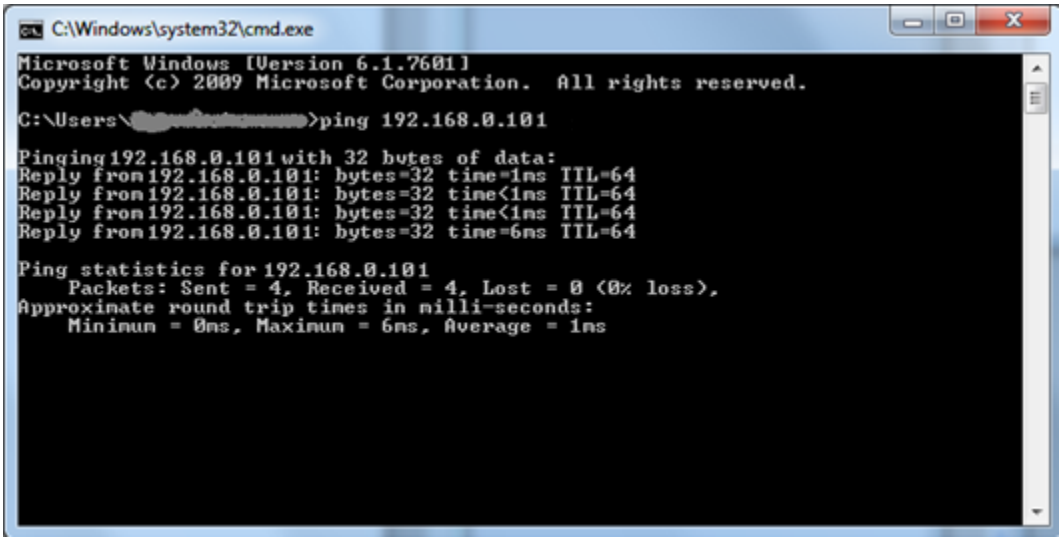
3.3.2 Optional: Ensure the Quark is accessible from the NAMT PC

Use ping to verify Versa Router™ 1000 Quark connectivity from the NAMT PC:

1. Click **Start**, **Run**, and type **cmd**.

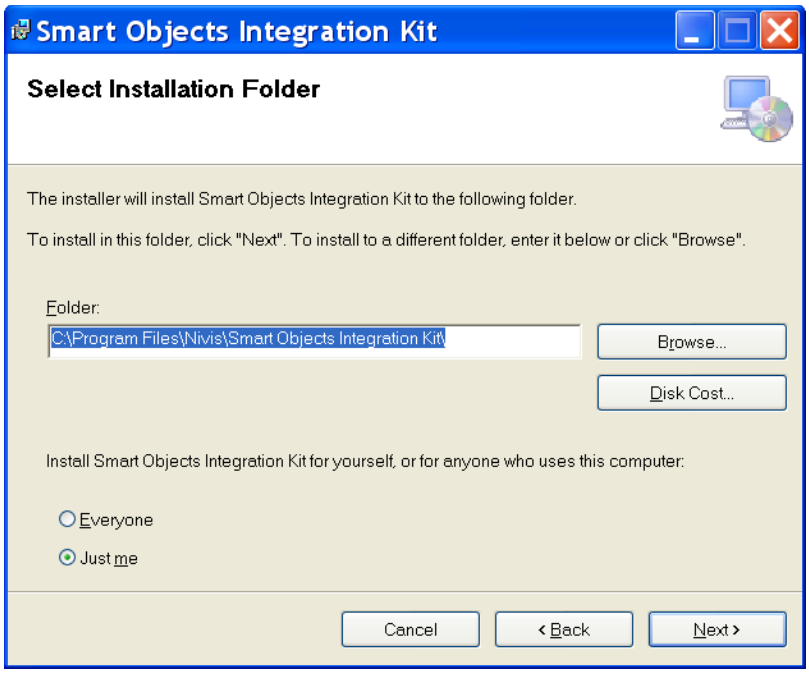


2. Type **ping 192.168.0.101**. The expected result is shown below.


C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\[redacted]>ping 192.168.0.101
Pinging 192.168.0.101 with 32 bytes of data:
Reply from 192.168.0.101: bytes=32 time=1ms TTL=64
Reply from 192.168.0.101: bytes=32 time<1ms TTL=64
Reply from 192.168.0.101: bytes=32 time<1ms TTL=64
Reply from 192.168.0.101: bytes=32 time=6ms TTL=64
Ping statistics for 192.168.0.101
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 6ms, Average = 1ms

3.3.3 Installing the NAMT

The NAMT comes with a standard application installer that allows a simple installation procedure. The installer can be launched by executing **setup.exe** in the install package. In the introduction screen, click **Next** to start the install procedure. The installer will automatically suggest **C:\Program Files\Nivis\Smart Objects Integration Kit** as the install folder.


Smart Objects Integration Kit
Select Installation Folder
The installer will install Smart Objects Integration Kit to the following folder.
To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".
Folder:
C:\Program Files\Nivis\Smart Objects Integration Kit\
Browse...
Disk Cost...
Install Smart Objects Integration Kit for yourself, or for anyone who uses this computer:
Everyone
Just me
Cancel
< Back
Next >

The user can change the path by entering or selecting the desired installation folder. The user can also specify whether the NAMT should be accessible to all users or only to the user installing it. After the corresponding installation options are selected, click **Next** to start the install process. When the setup is complete, click **Close** to finish the installation.

The application setup can be stopped at any time by clicking **Cancel**.

During the setup, a folder named "**Nivis**" will be created in the Windows Start menu. The NAMT can be launched or uninstalled by running the corresponding applications in this folder.

3.3.4 Starting up the NAMT

3.3.4.1 Overview

The user should wait several minutes after the Versa Router™ 1000 Quark powers on before attempting to connect NAMT to it.

Start the NAMT by double-clicking the "**Smart Objects Integration Kit**" icon on the desktop or Windows Start menu.

3.3.4.2 Connecting to a Versa Router™ 1000 Quark

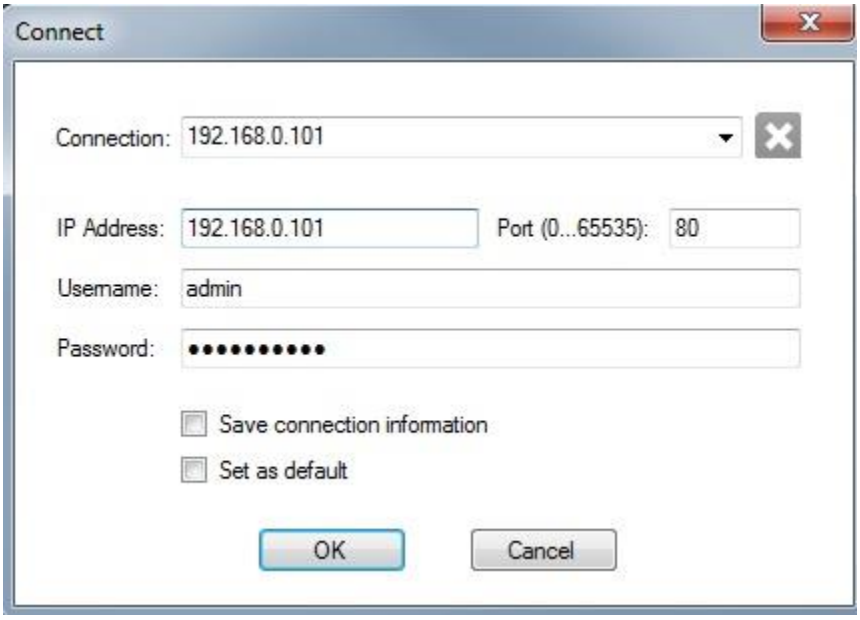
In order to connect the NAMT to the Versa Router™ 1000 Quark, the PC hosting the NAMT must be connected in the same network segment (or subnet) as the Versa Router™ 1000 Quark. If the Versa Router™ 1000 Quark is operating using the default network settings, it is recommended that the PC hosting the NAMT use the 192.168.0.201 IPv4 address.

See "Setting Up the Versa Router™ 1000 Quark" in the Quick Start Guide for matching the Versa Router™ 1000 Quark IP network settings with the network settings of the PC hosting the NAMT.

If the PC hosting the NAMT and the Versa Router™ 1000 Quark have IP addresses that are in the same network or subnet, the NAMT will connect to the Versa Router™ 1000 Quark when started.

At NAMT startup, the NAMT will ask for the information necessary for connecting to the Versa Router™ 1000 Quark:

- The Versa Router™ 1000 QuarkIPv4 (by default,192.168.0.101)
- Username (admin)
- Password (adminadmin)



The user must provide the IP address of the ER and the authentication credentials or select from a list of saved connections. When the user clicks **OK**, the NAMT will attempt to connect to the target ER. If the connection succeeds, the Network Monitoring module of the application will be started.

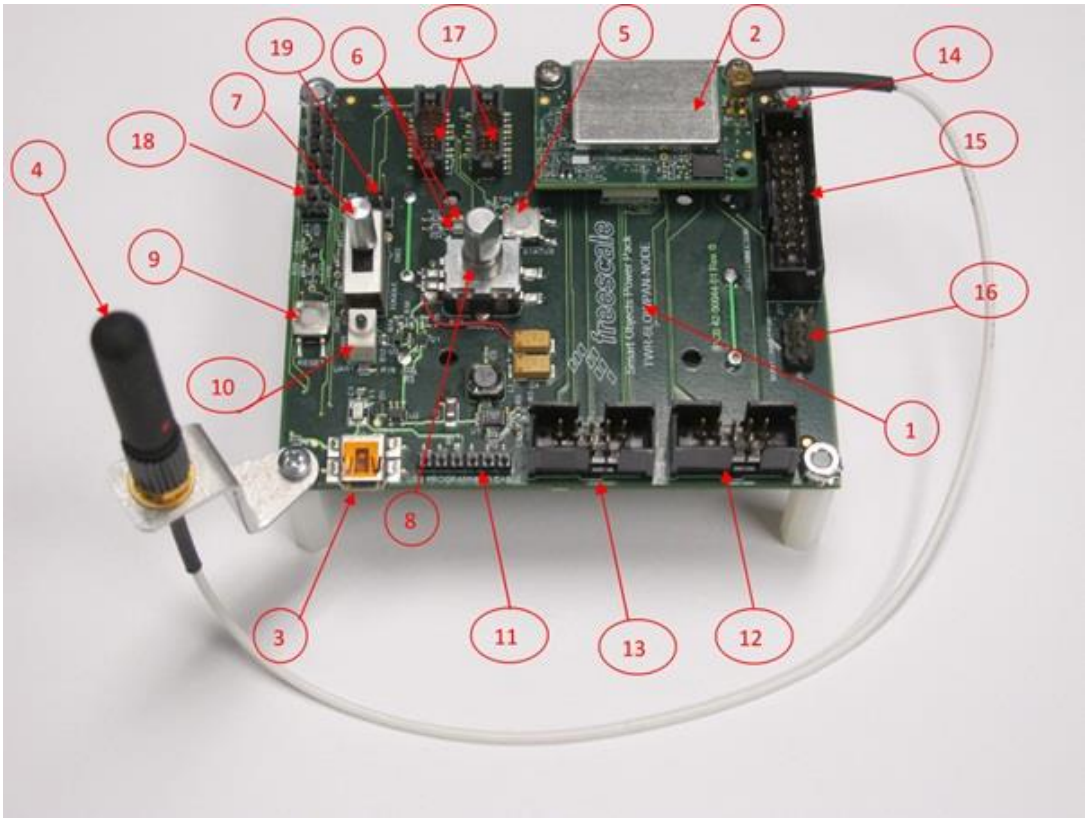
If the user clicks **Cancel**, the connection window will be closed and the main NAMT window will be displayed. The user will be able to connect to an ER, manage the application settings, or close the application.

4 Development Kit Components - Hardware

4.1.1 Smart Object Endpoint

4.1.1.1 General product view

The Smart Object Endpoint consists of a Power Pack unit (1) that has an SMO radio module (2) installed. The Smart Object Endpoint was designed to allow functionality as a line-powered endpoint or battery-powered endpoint. The Smart Object Endpoint can be powered by attaching either 3 AA batteries or a 5V DC mini-USB AC/DC adapter.



The following elements are included on the Smart Object Endpoint:

Nr.	Designator	Description
1	Power Pack	Power Pack board including all other elements for a Smart Object Endpoint
2	SMO Module	SMO Radio Module used for connecting to a wireless network
3	J2	Mini-USB connector for power input when 5V DC is used
4	ANT	Antenna and antenna cable that is attached to the SMO radio module
5	SW1	STATUS momentary push button for enabling the LED status display. The LED will display the unit registration status for only a limited period of time (~10 seconds) after the STATUS button is pressed. This is done to conserve power.
6	D3, D2	Two onboard LED's used as: - Green LED, D3, display network registration STATUS of the unit - Red LED, D2, used by the application
7	SW2	Slide switch used as a Digital Input sensor for demonstrations
8	R21	Potentiometer used as an Analog Input sensor for demonstrations
9	SW3	RESET momentary push button for hardware reset
10	SW4	Toggle switch selects between UART/SPI application interface used by API
11	J5	USB programming cable, header provides UART access to SMO Radio module
12	J17	UART Application connector, connects user application system via a serial port
13	J13	SPI Application connector, connects user application system via an SPI port
14	J18	Power jumper for SMO Radio module, used to access measurements of power consumption. Remove the jumper to conserve battery when the system is not in use.
15	J11	JTAG Connector used for K60 firmware updates
16	J9	BDM Connector used for MC13211 firmware updates
17	J3, J6	TWRPI connectors for attaching a Tower TWRPI Carrier Module
18	J8, J14, J15, J16, J17, J21	Configuration Jumpers allow connection/disconnection of various subsystems to the SMO Radio module pins.
19	J19, J20	LEDs connect/disconnect jumpers allow separation of LEDs from SMO Radio module pins.
20	BT1	Battery Holder installed on the bottom of the board (not shown in picture)

For full electrical specifications, please refer to the Power Pack and SMO Radio Module datasheet.

4.1.1.2 AC/DC adapter

The Smart Object Endpoint can be powered by 5V DC via the mini-USB J2 connector (3). The connector provides power only and cannot act as a USB data connection. Alternately, the power can be provided by any computer or device that has a USB host port. The equipment used for power should be able to provide a minimum of 3 Watts.

Manufacturer: Emerson Network Power

Manufacturer P/N: **DCH3-050US-0001**

Description: PLUG WLMNT 5V 3W US 2 MINI USB

Datasheet Link: https://www.powerconversion.com/assets/dch3_ds_us_1300949281.pdf

4.1.1.3 Antenna

4.1.1.3.1 Overview

The antenna conforms to SMO Radio module certifications and approvals. The Smart Object Endpoint uses a quarter wavelength antenna with +0 dib gain. Full antenna characteristics are provided in the manufacturer's datasheet.

4.1.1.3.2 902 – 928 MHz antenna specifications

Manufacturer: Antenna Factor (Linx Technologies)

Manufacturer P/N: **ANT-916-CW-RH-SMA**

Description: ANT 916 MHz, ¼ WAVE WHIP, RP-SMA (MALE)

Datasheet Link: http://www.linxtechnologies.com/resources/data-guides/ant-916-cw-rh.pdf

4.1.1.3.3 863 - 870 MHz antenna specifications

Manufacturer: Antenna Factor (Linx Technologies)

Manufacturer P/N: **ANT-868-CW-QW**

Description: ANT 868 MHz, ¼ WAVE WHIP, RP-SMA (MALE)

Datasheet Link: https://www.linxtechnologies.com/resources/data-guides/ant-868-cw-qw.pdf

4.1.1.4 Mini-USB power connection – J2

The mini-USB provides input connections for 5V as per USB standard. A USB Data connection is NOT available via this connector. Pin 1 is located on the left side of the connector.

Pin Number	Signal Name	Direction	Description
1	5V VCC	Power In	5V, 500mA power input (AC/DC adapter or computer USB port)
2	NC		Not Used
3	NC		Not Used
4	NC		Not Used
5	GND	Ground	GND - Ground

4.1.1.5 *USB to UART programming connection – J5*

The connection is intended for use with FTDI cable model TTL-232R-3V3; all signals are at a 3.3V TTL level. (http://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_TTL-232R_CABLES.pdf)

Pin Number	Signal Name	Direction	Description
1	GND	Ground	GND - Ground
2	USB_CTS	Output	CTS Signal of FTDI USB-UART cable
3	USB-VCC	Power Input	5V provided by FTDI USB-UART cable (not used)
4	USB_TXD_OUT	Input	TXD Signal of FTDI USB-UART cable
5	USB_RXD_IN	Output	RXD Signal of FTDI USB-UART cable
6	USB_RTS	Input	RTS Signal of USB-UART cable

4.1.1.6 *UART application connector – J17*

This connector provides access to the SMO radio module UART lines including flow control and LLWU wake-up capability. All signals are at a 3.3V TTL level.

Pin Number	Signal Name	Direction	Description
1	K11_RXD_IN	Input	K11 UART, RX Data Line
2	3V3	Power Output	Low Current 3V3 Output (used for level shifters)
3	K11_TXD_OUT	Output	K11 UART, TX Data Line
4	3V3	Power Output	Low Current 3V3 Output (used for level shifters, same as pin 2)
5	nUART1_CTS	Input	K11 UART, CTS Flow Control Line
6	GND	Ground	GND - Ground
7	nUART1_RTS	Output	K11 UART, RTS Flow Control Line
8	GND	Ground	GND - Ground
9	LLWU_P13	Input	K11 LLWU_P13 Wake-up line (internal pull-up)
10	GND	Ground	GND - Ground

NOTE Select UART position for SW4 in order to use this connection.

4.1.1.7 SPI application connector – J13

This connector provides access to SMO radio module Master Mode SPI lines and LLWU wake-up capability. All signals are at a 3.3V TTL level.

Pin Number	Signal Name	Direction	Description
1	SPI1_PCS0	Output	Chip Select Line (Master Mode= Output)
2	3V3	Power Output	Low Current 3V3 Output (use for level shifters)
3	SPI1_SIN	Input	MISO SPI Data Input Line (Master Mode)
4	3V3	Power Output	Low Current 3V3 Output (use for level shifters)
5	SPI1_SOUT	Output	MOSI SPI Data Output Line (Master Mode)
6	GND	Ground	GND - Ground
7	SPI1_SCK	Output	SPI Clock Line Output (Master Mode)
8	GND	Ground	GND - Ground
9	LLWU_P13	Input	K11 LLWU_P13 Wake-up line (set Low for wake-up)
10	GND	Ground	GND - Ground

NOTE SPI Feature will be supported in future firmware; select SPI on SW4 in order to use this feature.

4.1.1.8 TWRPI application connector – J3, J6

This connector implements a TWRPI interface for use with selected Tower TWRPI adapters. Signals are a subset of current TWRPI standard implementation. All signals are at a 3.3V TTL level.

J3 Pin Number	Signal Name	Direction	Description
1	GND	Ground	GND - Ground
2	GND	Ground	GND - Ground
3	I2C_SCL	Output	I2C Bus Clock Pin
4	I2C_SDA	Bidirectional	I2C Bus Data Pin
5	GND	Ground	GND - Ground
6	GND	Ground	GND - Ground
7	GND	Ground	GND - Ground
8	GND	Ground	GND - Ground
9	SPI1_SIN	Input	MISO SPI Data Input Line (Master Mode)
10	SPI1_SOUT	Output	MOSI SPI Data Output Line (Master Mode)
11	SPI1_PCS0	Output	Chip Select Line (Master Mode= Output)
12	SPI1_SCK	Output	SPI Clock Line Output (Master Mode)
13	GND	Ground	GND - Ground
14	GND	Ground	GND - Ground
15	LLWU_P13	Input/Output	General Use as Digital I/O
16	BOOT_SW	Input/Output	General Use as Digital I/O
17	UART1_TXD	Output	K11 UART1, TX Data Line
18	UART1_RXD	Input	K11 UART1, RX Data Line
19	nUART1_RTS	Output	K11 UART1, RTS Flow Control Line
20	nUART1_CTS	Input	K11 UART1, CTS Flow Control Line

J6 Pin Number	Signal Name	Direction	Description
1	NC		NOT USED
2	3V3	Power Output	Low Current 3V3 Output
3	GND	Ground	GND - Ground
4	NC		NOT USED
5	GND	Ground	GND - Ground
6	GND	Ground	GND - Ground
7	GND	Ground	GND - Ground
8	ADC_SE12	Analog Input	Analog Input Single-Ended to K60
9	SPI1_SIN	Input	MISO SPI Data Input Line (Master Mode)
10	SPI1_SOUT	Output	MOSI SPI Data Output Line (Master Mode)
11	SPI1_PCS0	Output	Chip Select Line (Master Mode= Output)
12	NC		NOT USED
13	GND	Ground	GND - Ground
14	GND	Ground	GND - Ground
15	GND	Ground	GND - Ground
16	GND	Ground	GND - Ground
17	NC		NOT USED
18	NC		NOT USED
19	GND	Ground	GND - Ground
20	PTA5	Input/Output	General Use as Digital I/O

4.1.1.9 *Configuration jumpers*

Configuration jumpers allow disconnection of various board subsystems from the SMO radio module pins. They are intended for separation for consumption measurement purposes or to allow alternate usage of the pins when the TWRPI connection is used.

Jumper	Usage	Description
J14	Analog Disconnect	Connects/Disconnects R21 analog voltage from the SMO Radio
J15	USB_CTS Disconnect	Connects/Disconnects USB_CTS signal from the SMO Radio
J8	USB_TXD_OUT Disconnect	Connects/Disconnects USB_TXD_OUT signal from the SMO Radio
J16	USB_RTS Disconnect	Connects/Disconnects USB_RTS signal from the SMO Radio
J21	SW2 Disconnect	Connects/Disconnects SW2 signal from the SMO Radio
J7	USB_RXD_IN Disconnect	Connects/Disconnects USB_RXD_IN signal from the SMO Radio
J10	Battery Measurement Disconnect	Connects/Disconnects battery measurement from SMO Radio
J19	LED D2 Disconnect	Connects/Disconnects LED D2 (RED) from the SMO Radio
J20	LED D3 Disconnect	Connects/Disconnects LED D3 (GREEN) from the SMO Radio

4.1.1.10 JTAG connector – J11

This connector implements standard ARM JTAG 20 pin connection for K60 processor use.

Pin Number	Signal Name	Direction	Description
1	3V3	Power Output	Low Current 3V3 Output (for level shifters)
2	NC		NOT USED
3	NC		NOT USED
4	GND	Ground	GND - Ground
5	JTAG_TDI	Input	JTAG Signal TDI Data Input
6	GND	Ground	GND - Ground
7	JTAG_TMS	Input	JTAG Signal TMS Mode Select
8	GND	Ground	GND - Ground
9	JTAG_TCLK	Input	JTAG Signal TCLK Clock Input
10	GND	Ground	GND - Ground
11	NC		NOT USED
12	GND	Ground	GND - Ground
13	JTAG_TDO	Output	JTAG Signal TDO Data Output
14	GND	Ground	GND - Ground
15	nRESET_K11	Input	JTAG Signal RESET processor reset
16	GND	Ground	GND - Ground
17	NC		NOT USED
18	GND	Ground	GND - Ground
19	NC		NOT USED
20	GND	Ground	GND - Ground

4.1.1.11 BDM connector – J9

This connector implements Freescale BDM connection for programming MC12311.

Pin Number	Signal Name	Direction	Description
1	BKGD	Bidirectional	BDM Signal BKGD background mode
2	GND	Ground	GND - Ground
3	NC		NOT USED
4	nRESET_MC	Input	BDM Signal RESET processor reset
5	NC		NOT USED
6	3V3	Power Output	Low Current 3V3 Output (for level shifters)

4.1.2 Versa Router™ 1000 Quark hardware description

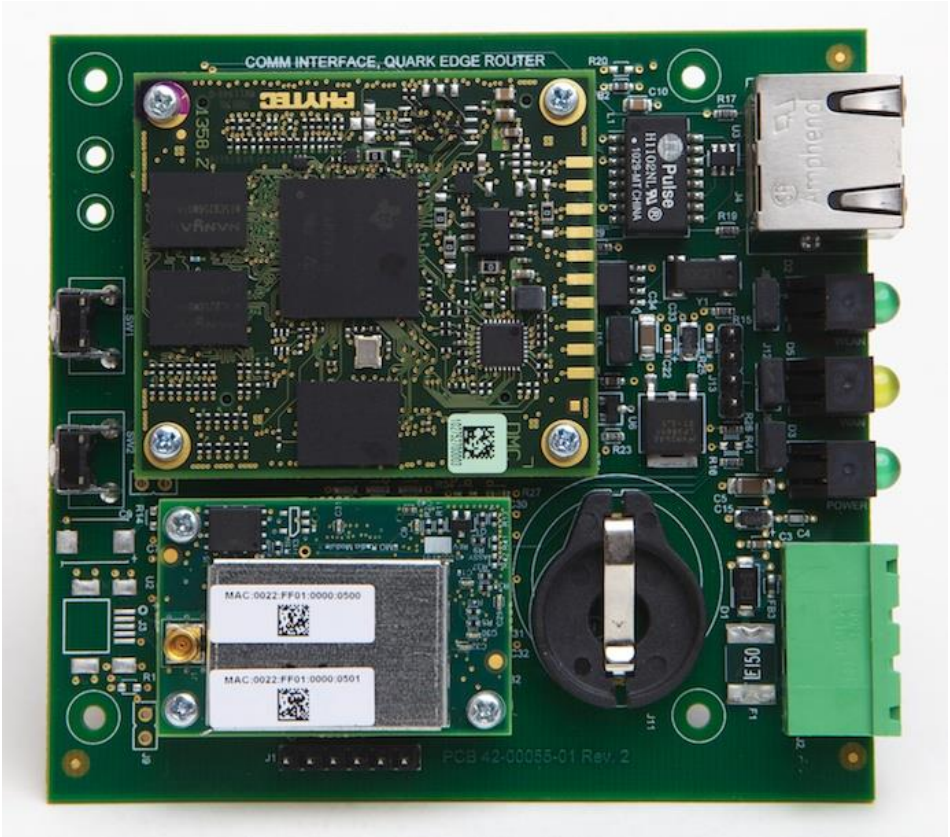
4.1.2.1 General product view

The Versa Router™ 1000 Quark is running on Phytec designed and built phyCORE-AM335x, built around the AM335x processor with the functionality of a network router. It requires only an Ethernet connection and a power adapter in order to run the hardware. The power is provided via an AC/DC adapter. The Smart Object's wireless connection is established via the on-board radio connected to the antenna connector on the case.

See the VR1000 datasheet on the Nivis website www.nivis.com for additional details.

Nivis provides both standard and OEM versions of the product. The standard version is basically the OEM version enclosed in a custom aluminum enclosure, with antenna and power supply.

The OEM version does not have the enclosure, antenna, or power supply.



4.1.2.2 Connectors, buttons, LEDs

The pictures below show the connectors, buttons, and LEDs existing on the VR1000 faceplate(s).

The green connector (1) in the left is for power supply connection.

There are 3 main LEDs on the Versa Router™ 1000 Quark faceplate: “POWER,” “WAN,” and “WLAN.” The other two secondary LEDs are the ETH link/activity LEDs.

The “POWER” LED (2) lights when the power is applied to the Versa Router™ 1000 Quark.

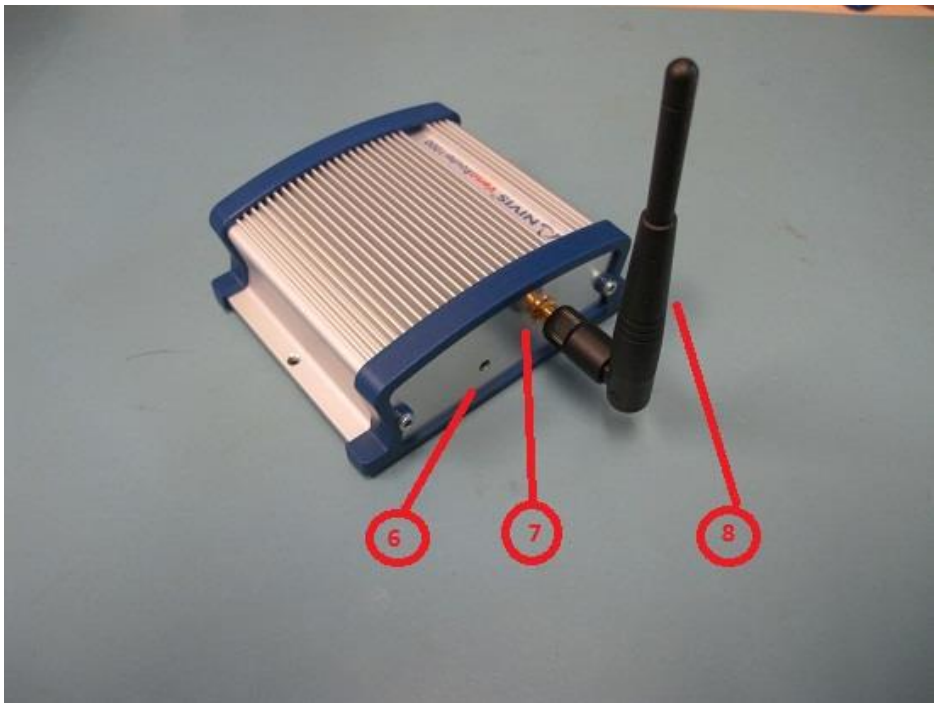
The “WAN” LED (3) lights when there is a NAMT connected to the Versa Router™ 1000 Quark.

The “WLAN” LED (4) lights when the Router has activity over RF (when the Transceiver transmits). A secondary feature of the “WLAN” LED is blinking when the user resets the Quark to defaults.

To the right is the RJ-45 connector (5) for WLAN connectivity.

The “Reset to Defaults” button (6) on the rear faceplate serves to reset the settings to the factory defaults. See the following section for additional details.

Also on the rear faceplate is the antenna connector (7). The antenna is shown connected (8).



4.1.2.3 *The Quark "Reset to Default" button*

If the Quark network settings(for example, the IP) are lost or if for any reason the configuration becomes non-functional, you can use the "Reset to defaults" button to reset all application configurations and Quark Network configurations.

To do this, press and hold the "Reset to Defaults" button (located on the back faceplate opposite the ETH/Power faceplate). Depress the button for up to 10 seconds, until the "WLAN" LED starts blinking. At that time, the Quark connectivity settings and the application configuration are reset to factory defaults. The LED will keep blinking for 10 seconds, then the board will perform a reboot.

NOTE After the reboot, the Quark network configuration will be: IPv4 **192.168.0.101**, network mask **255.255.255.0**, gateway **192.168.0.1.**

4.1.2.4 *Power supply specifications*

The DC power supply used for the Versa Router™ 1000 Quark is included with the Smart Objects Integration Kit.

The power supply will provide 25 Watts of power.

Manufacturer: **SCEPTRE**

Manufacturer P/N: **PS2D-5050APL05**

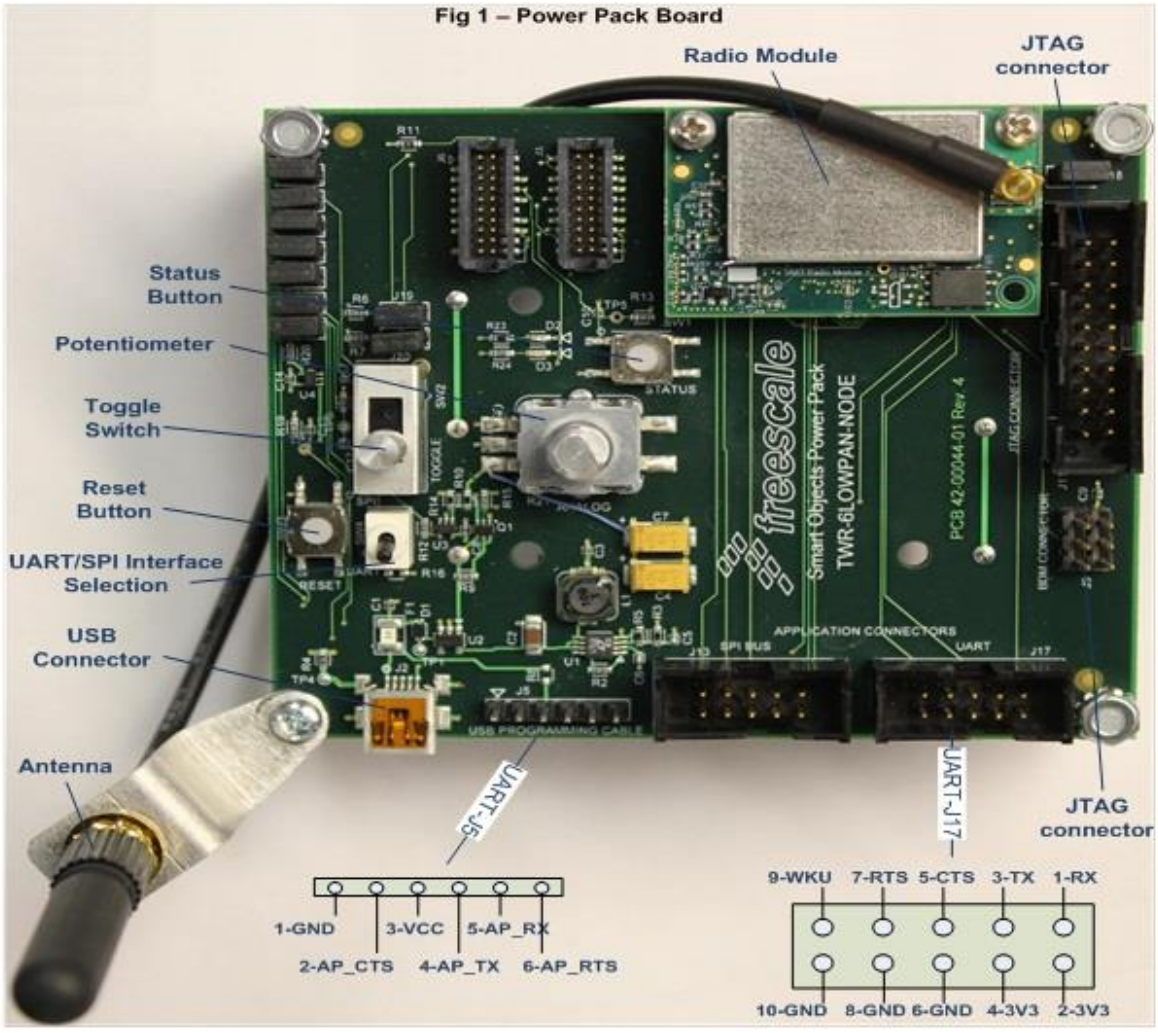
Description: ADAPTER WALLMOUNT 5V DC OUT, 2A (25 WATT)

Datasheet Link: https://www1.elfa.se/data1/wwwroot/assets/datasheets/std-0505p21_eng_tds.pdf

5 Development Kit Components - Software

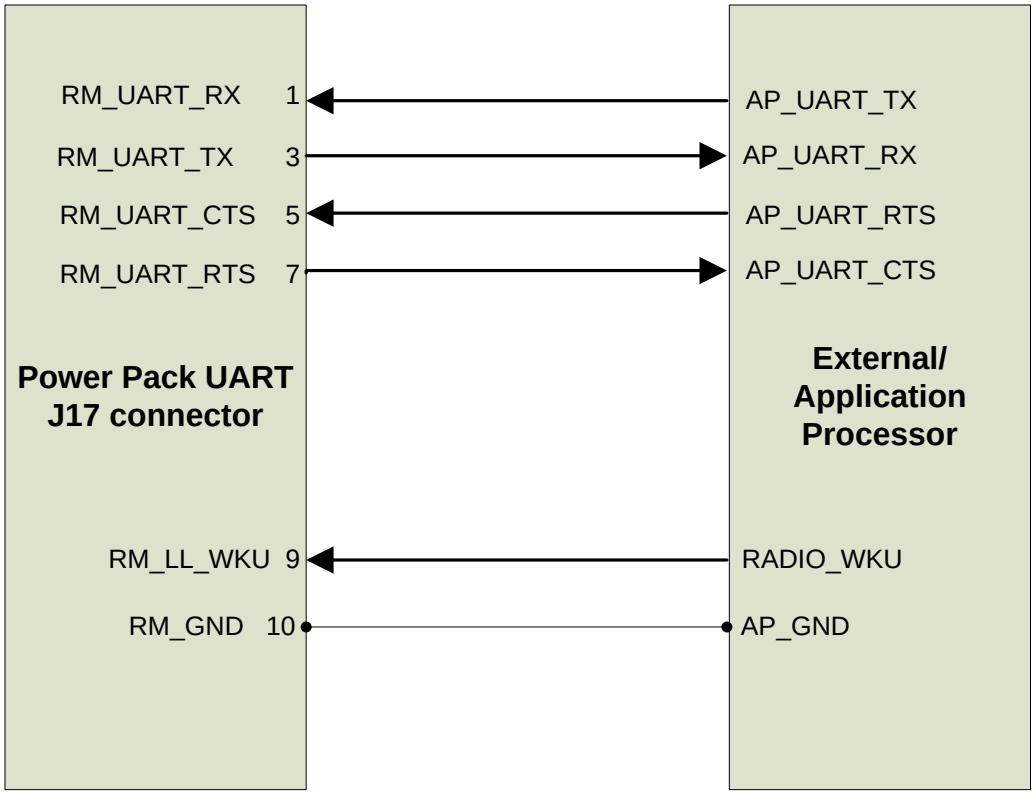
5.1.1 The Application Processor Interface (API)

The Power Pack board was designed to allow easy interfacing with the Nivis Smart Object radio module, offering integrators access to the main HW interfaces. Please refer to the picture below for the main components on the Power Pack board.



The connector used for UART communication with the application processor is J17 (the arrow indicates pin 1).

The picture below shows the pins for UART interface from the Smart Objects Radio Module to their equivalent signals on the application processor. The pins on the UART-J5 header are connected to the corresponding pins on the UART-J17 header and can be easily probed during integration (e.g., logic analyzer). The RADIO_WKU signal can also be probed on pin 9 on the SPI connector.



For additional details on connecting the SO with an external application processor, please refer to the API document “Nivis Smart Object API Integration Manual rev 2.4”

The API allows an external application processor to get information on the HW platform, read the time, request network-related information (join status, link quality, RPL info), and manage resources (list, read, write) and UDP-related commands (socket and datagram management).

For a list of detailed API messages and descriptions, please see the API document “Nivis Smart Object API Integration Manual rev 2.4”

5.1.2 The Versa Router™ 1000 Quark Software

5.1.2.1 HTTP-COAP proxy &XML -EXI translator

The HTTP-COAP proxy and XML-EXI translator is a translator from XML over HTTP to EXI over COAP and a proxy for COAP resources.

The client apps can query directly the COAP resources on the Smart Objects using EXI on top of COAP; however, this is not recommended, as it may exceed with ease the bandwidth available.

We recommend sending the queries as XML on top of HTML. The proxy on the Versa Router™ 1000 Quark will serve the resource from the cache or query the SO if necessary; at the same time, it will translate from EXI/COAP to XML/HTTP. If configured, the proxy will subscribe to observable resources on the Smart Objects, further reducing the bandwidth requirements.

Advantages of using a XML/HTTP – EXI/COAP proxy against interrogating the Smart Objects directly include:

- Controllable bandwidth requirements
- Reduced bandwidth requirements ensured through use of caching on the proxy
- Reduced bandwidth requirements ensured through use of subscription to observable resources from the proxy
- Fast development by using XML/HTTP

5.1.2.1.1 COAP – HTTP proxy interface description

The proxy interface allows a client application to list the observable resources, get the device list and topology, and observe resources (including the configuration of the resources observed).

5.1.2.1.2 HTTP status codes

The CoAP – HTTP status code translation is performed according to the following table.

CoAP Code	HTTP Status Code
65	**201** Created
66	**204** No Content
67	**304** Not Modified
68	**204** No Content
69	**200** OK
128	**400** Bad Request
129	**401** Unauthorized
130	**400** Bad Request
131	**403** Forbidden
132	**404** Not Found
133	**405** Method Not Allowed
134	**406** Not Acceptable
140	**412** Precondition Failed
141	**413** Request Entity Too Large
143	**415** Unsupported Media Type
160	**500** Internal Server Error
161	**501** Not Implemented
162	**502** Bad Gateway
163	**503** Service Unavailable
164	**504** Gateway Timeout
165	**505** HTTP Version Not Supported

While most HTTP status codes returned are a direct translation from the CoAP corresponding response code, there are several error codes that may be sent by the proxy.

HTTP Request	HTTP Error Code	Details
Any request	**504** Gateway Timeout	The request could not be handled in the configured time interval. **Response string**: “Gateway Timeout”
	501 Not Implemented	The request could not be parsed. **Response string**: “Can't parse Http request”
	400 Bad Request	The HTTP request is not valid according to intended functionality as per the HttpCoapProxy design document and therefore cannot be handled. **Response string**: “Invalid request”
Device resource request	**501** Not Implemented	The request could not be sent over the CoAP interface at this time. **Response string**: “Can't SendRequestToCoap”
		The requested resource could not be created locally. **Response string**: “Can't AddReqRspResource”
	502 Bad Gateway	Decoding error. **Response string**: “Can't convert Coap response to Http response, Exi to Xml failed”
	409 Conflict	A *no-cache* HTTP request was made for an observed resource. Requests for observed resources may not bypass the cache and such a request should not be attempted. **Response string**: “Can't bypass cache for an observed resource”
	504 Gateway Timeout	There was no response from the CoAP interface in the configured timeout interval. **Response string**: “Gateway Timeout”
		Another request for the same resource was already made and has not yet been answered. **Response string**: “Another request for this resource was sent before”
		An observable resource is in the subscribing process, and thus a new request for the resource will not be made and data will only be available once the subscription is established. **Response string**: ”Resource is in subscription process”

HTTP Request	HTTP Error Code	Details
Observe resources request	**400** Bad Request	The XML observation request was badly formed. **Response string**: "Invalid query"
		The individual observable element request was badly formed. **Response string**: "Observing resources request is bad formed"
		The XML observation request is not a PUT request. **Response string**: "Request is bad formed, invalid method, it should be PUT"
		The XML observation request content type is invalid. **Response string**: "Request is bad formed, invalid content type, it should be application/xml"
		The XML observation request content is invalid. **Response string**: "Request is bad formed, invalid Xml content." **Response string**: "Request is bad formed, error processing Xml Observation List elements" **Response string**: "Request is bad formed, Xml Observation List invalid finish"
Storage data retrieval request	**400** Bad Request	The request query has invalid or conflicting elements. **Response string**: "Storage-data-get request is bad formed"
Device list request	**400** Bad Request	The device list search query is badly formed. **Response string**: "Invalid query"

5.1.2.1.3 Communicating with HTTP CoAP Proxy

If the HTTP Client is a web-browser, the HTTP client has to manually set the proxy configuration, by setting the Proxy IP to the value of Quark IP, and the port to 9999.

EXAMPLE:

Considering CoAP server [2001::1:101]

Considering Quark (HttpCoapProxy): 10.32.0.160, listens by default on TCP port 9999

The client web browser must be configured for "Manual proxy configuration":

HTTP Proxy: 10.32.0.160 Port: 9999

5.1.2.1.4 Resources

HTTP query: http://resources

HTTP response:

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE ResourceList [
<!ELEMENT ResourceList  (CoAPResource)>
<!ELEMENT CoAPResource  (Value)>
<!ELEMENT Value  (#PCDATA)>
<!ATTLIST Value Name CDATA #IMPLIED>
<!ATTLIST Value Type CDATA #IMPLIED>
]>
<ResourceList>
        <CoAPResource>
                <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                <Value Name=’CoRELinkFormat’ Type=’String’>…</Value>
        </CoAPResource>
        <CoAPResource>
                <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                <Value Name=’CoRELinkFormat’ Type=’String’>…</Value>
        </CoAPResource>
        <CoAPResource>
                <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                <Value Name=’CoRELinkFormat’ Type=’String’>…</Value>
        </CoAPResource>
        …
</ResourceList>
```

5.1.2.1.5 Device list

HTTP query: http://device-list(?search*)

Where search can be: MAC=HHHHHHHHHHHH or IPv6Addr=HHHHHHHHHHHHHHHHHHHHHHHH

HTTP response:

```
<?xml version='1.0' encoding='ascii'?>
<!DOCTYPE DeviceList [
<!ELEMENT DeviceList  (Device)>
<!ELEMENT Device  (Value)>
<!ELEMENT Value  (#PCDATA)>
<!ATTLIST Value Name CDATA #IMPLIED>
<!ATTLIST Value Type CDATA #IMPLIED>
]>

<DeviceList>
        <Device>
                <Value Name='MAC' Type='BinHex'>…</Value>
                  <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                <Value Name='DeviceType' Type='Number'>…</Value>
                <Value Name='Status' Type='Number'>…</Value>
                <Value Name='LastComm' Type='Number'>…</Value>
                <Value Name='LastRegistration' Type='Number'>…</Value>
        </Device>
        <Device>
                <Value Name='MAC' Type='BinHex'>…</Value>
                  <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                <Value Name='DeviceType' Type='Number'>…</Value>
                <Value Name='Status' Type='Number'>…</Value>
                <Value Name='LastComm' Type='Number'>…</Value>
                <Value Name='LastRegistration' Type='Number'>…</Value>
        </Device>
        …
```

</DeviceList>

NOTES

- LastComm – UTC time in seconds (since 1970) – time when last message (network or APP) was received from device
- LastRegistraton - UTC time in seconds (since 1970) – time when last DAO message was received from device
- DeviceType: { 0:Any; 1:TypeNMS; 2:reserved; 3:TypeER; 4:reserved; 5:reserved; 6:TypeEndpointRPL; 7:TypeRplRoot; 8:TypeRplLocalEndpointIfUp; 9:TypeRplLocalEndpointIfDown; 10:TypeRplEndpointOnBattery; 11: TypeRplSimEndpoint}

5.1.2.1.6 Topology

HTTP query: http://topology

HTTP response:

```
<?xml version='1.0' encoding='ascii'?>
<!DOCTYPE Topology [
<!ELEMENT Topology  (Device)>
<!ELEMENT Device  (Value, Parents)>
<!ELEMENT Parents  (Parent)>
<!ELEMENT Parent  (Value)>
<!ELEMENT Value  (#PCDATA)>
<!ATTLIST Value Name CDATA #IMPLIED>
<!ATTLIST Value Type CDATA #IMPLIED>
]>

<Topology>
        <Device>
                <Value Name='MAC' Type='BinHex'>…</Value>
                  <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                  <Parents>
                        <Parent>
                                <Value Name='MAC' Type='BinHex'>…</Value>
                                   <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                                <Value Name='ExpirationTime' Type='Number'>…</Value>
                                <Value Name='PathSequence' Type='BinHex'>…</Value>
                                <Value Name='PathControl' Type='BinHex'>…</Value>
                                  <Value Name='Preferred' Type='Bool'>…</Value>
                        </Parent>
                        <Parent>
                                <Value Name='MAC' Type='BinHex'>…</Value>
                                   <Value Name='IPv6Addr' Type='BinHex'>…</Value>
                                <Value Name='ExpirationTime' Type='Number'>…</Value>
```

```
                    <Value Name='PathSequence' Type='BinHex'>…</Value>
                    <Value Name='PathControl' Type='BinHex'>…</Value>
                      <Value Name='Preferred' Type='Bool'>…</Value>
                </Parent>
                …
            </Parents>
        </Device>
        …
</Topology>
```

5.1.2.1.7 Resource requests

An HTTP request for a target resource is translated into a CoAP request and then sent to the appropriate device.

The HTTP method (GET/PUT/POST/DELETE) is translated in the equivalent CoAP method.

The HTTP options are, when possible, translated into CoAP options. Otherwise they are ignored, with the exception of the cache-control options ("***Pragma: no-cache***" /"***Cache-Control: no-cache***"), which may be used to indicate cache bypass to the proxy itself and otherwise has no CoAP equivalent.

The HTTP payload is translated as CoAP payload.

HTTP request: http://[device_ipv6]/resource_path(?cmd=*)

where the optional cmd query may be as per device specification.

HTTP response:

- If the request was successful, the response will contain the device response and the corresponding HTTP status codes.
- If the request could not be sent for any reason, the response will contain an appropriate status and an informative message.

EXAMPLE

http://[2001::1:101]/

http://[2001::1:101]/app/ptm

5.1.2.1.8 Observing resources

5.1.2.1.8.1 Overview

HttpCoapProxy Observation List - list of resources ([device ipv6 addr]/resource_path) which were processed by HttpCoapProxy for putting under observation.

HttpCoapProxy Observation Cache - list of resources which are put under observation by HttpCoapProxy at current moment, with their corresponding responses.

5.1.2.1.8.2 Get observation list

HTTP query : http://observe-resources-get(?search*)

where search can be: MAC=HHHHHHHHHHHH or IPv6Addr=HHHHHHHHHHHHHHHHHHHHHHHH

HTTP response:

List of resources which are are put under observation by HttpCoapProxy at that moment.

This is actually the HttpCoapProxy Observation List in XML format.

```
<ObservationList>
    <Resource>
        <Value Name='DeviceMAC' Type='BinHex'>…</Value>
        <Value Name='DeviceIPv6Addr' Type='BinHex'>…</Value>
        <Value Name='UriPath' Type='String'>…</Value>
        <Value Name='State' Type='String'>…</Value>
<Value Name='PubPeriod' Type='Number'>…</Value>
    </Resource>
    <Resource>
        <Value Name='DeviceMAC' Type='BinHex'>…</Value>
        <Value Name='DeviceIPv6Addr' Type='BinHex'>…</Value>
        <Value Name='UriPath' Type='String'>…</Value>
<Value Name='State' Type='String'>…</Value>
<Value Name='PubPeriod' Type='Number'>…</Value>
    </Resource>
    …
</ObservationList>
```

NOTES

The “State” field can take the following values:

- “Started”
- “Can’t send request”
- “Device not in Device List”
- “Waiting for response from device”
- “Device didn't respond during timeout”
- “Resource doesn't exist”
- “Resource is not observable”
- “Subscribed”
- “Unsubscribed”

The PubPeriod field value is a number that describes the publication period set by the user for that observed resource. If “0”, it means that publication period for that resource is the default one, set by the device, and can be found in the ResourceDiscovery report (http://resources). If not “0”, it means that the publication period was explicitly set by the user to that value.

5.1.2.1.8.3 Enable observation for target resource

HTTP query: http://observe-resources-set?enable=1&device=device_ipv6_addr&path=resource_path

HTTP response:

- If the request is well formed, a 200 "OK" "Observation request for SUBSCRIPTION passed forward" message will be retrieved to the client. This is a sign that the request was processed and HttpCoapProxy will subscribe for observing the specified resource.
- If the request is not well formed, a 400 “Bad Request" response will be retrieved.

Every time a client makes a request for a resource, http://[device ipv6 addr]/resource_path, HttpCoapProxy will look for it in its Observation Cache:

- If it is found, it will be retrieved to the client, and it means that the resource is observable and that the client or another client made an earlier request to put it under observation.
- If that resource does not exist in the HttpCoapProxy Observation Cache, the normal flow will be followed: Send Coap request, wait for Coap response, translate to Http, and send back to Http client.

NOTE

http://observe-resourcesset?enable=1&device=device_ipv6_addr&path=resource_path &pb=publish_time can be used to determine the observed resource to be publish every “publish_time” seconds.

A successful request will result in assigning the observation state “Started” for the target resource (see description in Get observation list). Once a resource is in the “Subscribed” state, it is assumed that periodic notifications will be received for it, often enough so that the proxy will always have valid cached data available (data is valid for a max-age interval as indicated by the CoAP notification, or until newer data is received). If the max-age time interval passes and there are no new notifications, the CoAP subscription may be automatically renewed depending on the configurable option ALLOW_RESUBSCRIPTION_ON_TIMEOUT (option is set to NO by default, meaning subscription is NOT renewed).

5.1.2.1.8.4 Disable observation for target resource

HTTP query: http:// observe-resources-set?disable=1&device=device_ipv6_addr&path=resource_path

HTTP response:

- If the request is well formed and the resource is found in the Observation List, a 200 "OK" “Observation request for UNSUBSCRIPTION passed forward" message will be retrieved to the client. This is a sign that the request was processed and the HttpCoapProxy will unsubscribe from observing specified resource.
- If the request is well formed, but the resource is not found in the Observation List, a 200 "OK" "Observation request for UNSUBSCRIPTION wasn't passed forward, maybe resource is not in observation list" message will be retrieved to client.
- If the request is not well formed, a 400 “Bad Request" response will be retrieved.

5.1.2.1.8.5 Set observation for multiple resources

HTTP PUT query: http://observe-resources-set-xml

Where query content is an XML file with following format:

```
<ObservationListSet>
        <Resource>
            <Value Name='DeviceIPv6Addr' Type='BinHex'>…</Value>
            <Value Name='UriPath' Type='String'>…</Value>
            <Value Name='Action' Type='String'>…</Value>
        </Resource>
        <Resource>
            <Value Name='DeviceIPv6Addr' Type='BinHex'>…</Value>
            <Value Name='UriPath' Type='String'>…</Value>
            <Value Name='Action' Type='String'>…</Value>
        </Resource>
        …
</ObservationListSet>
```

NOTE

Action - is a string which defines the type of action that HttpCoapProxy should take on the specified resource (Enable – set as observable, HttpCoapProxy will subscribe for that resource, Disable – unset, HttpCoapProxy will unsubscribe from observing that resource)

HTTP response :

- If XML file is well formed then a 200 "OK" "XML observation list processed" will be sent back to client;
- If XML file is not well formed, 400 "Bad Request" response will be retrieved.

5.1.2.1.8.6 Retrieving data history

HTTP query: http://storage-data-get/(?device=device_ipv6_addr&since=time_since&until=time_until &oldest/newest=N/path=resource_path)

- Query may specify “device”
- Query may specify “since”, “until”, both of them or none.
- If query specified device one of “oldest”, “newest” queries may be specified.
- Query may specify a “path” containing a target resource path.

HTTP response:

- If request is well formed, HttpCoapProxy will retrieve an XML with history data which is stored for *device_ipv6_addr* device, from time_since and until *time_until*; *time_since* and time_until must be specified in UNIX timestamp format. If both “since” and “until” queries are missing, will be retrieved an XML with all history data which is stored for *device_ipv6_addr* device; this initial XML report will contain a number of X records. If “oldest” query is specified, the XML report will contain only first *N* records from X (or X if *N* >X), which represent the oldest records from the initial report. If “newest” query is specified, the XML report will contail only last *N* records from X (or X if *N* >X), which represent the newest records from the initial report;
- If path is specified, the proxy will perform a search using the provided path filter.
- If no *device_ipv6_addr* is provided, the proxy will provide the history for the entire storage.
- If no *device_ipv6_addr* is provided, “oldest” and “newest” queries do not apply.
- If storage for *device_ipv6_addr* device doesn’t exist, “Storage not found” response will be retrieved;
- If request is not well formed, 400 "Bad Request" response will be retrieved, with the information “Storage-data-get request is bad formed”.

Each record from the response XML has the following form:

```
<Record>
   <uriPath>Path</uriPath>
   <e t="timestamp"><Application v="value"/></e>
</Record>
```

If device is provided, the response XML will have the following form:

```
<DataRetrieving>
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
```

```
    …
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
</DataRetrieving>
```

If device is not provided, the response will have the following form:

```
<DataRetrieving>
<Device uriHost=device_ip>
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
    …
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
<Device>
…
<Device uriHost=device_ip>
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
    …
    <Record>
        <uriPath>path</uriPath>
        <e t="timestamp"><Application v="value"/></e>
    </Record>
<Device>
</DataRetrieving>
```

EXAMPLES of requests

http://storage-data-get/?device=2001::3&since=1358438082&until=1358438277

http://storage-data-get/?device=2001::3&since=1358438082&until=1358438277&oldest=5

http://storage-data-get/?device=2001::3&newest=10

http://storage-data-get/?device=2001::3& path=/app/sw

http://storage-data-get/?path=/app/sw

http://storage-data-get/

EXAMPLE of a record in response XML:

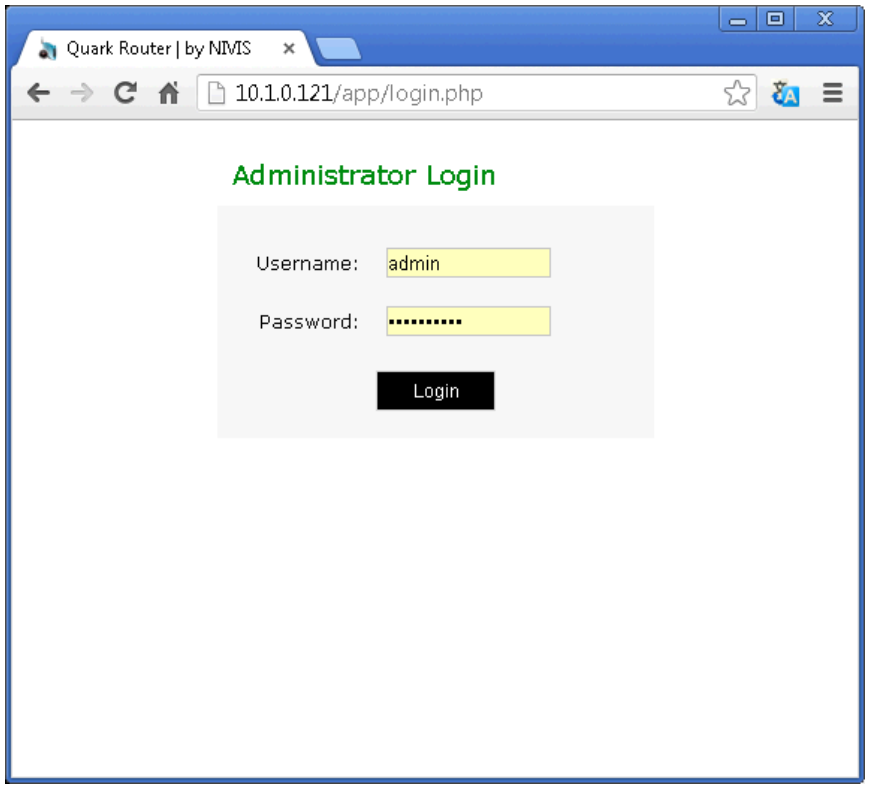
```
<Record>
   <uriPath>/app/sw</uriPath>
   <e t="1358831522"><Sw v="1"/></e>
</Record>
```

5.1.2.2 Configuration and administration of the VersaRouter™ 1000

5.1.2.2.1 Overview

Web-based administration is the *preferred* method for administration/configuration of the Versa Router™ 1000 Quark. It requires a web browser and the IP of the Versa Router™ 1000 Quark. The Versa Router™ 1000 Quark must be connected to the local LAN and then powered on, and the IP/mask of the router must be accessible from the PC where the browser is running.

The administration website for the Versa Router™ 1000 Quark can be accessed by pointing a web browser to the Quark IP.



Login with username: **admin** and password: **adminadmin**.

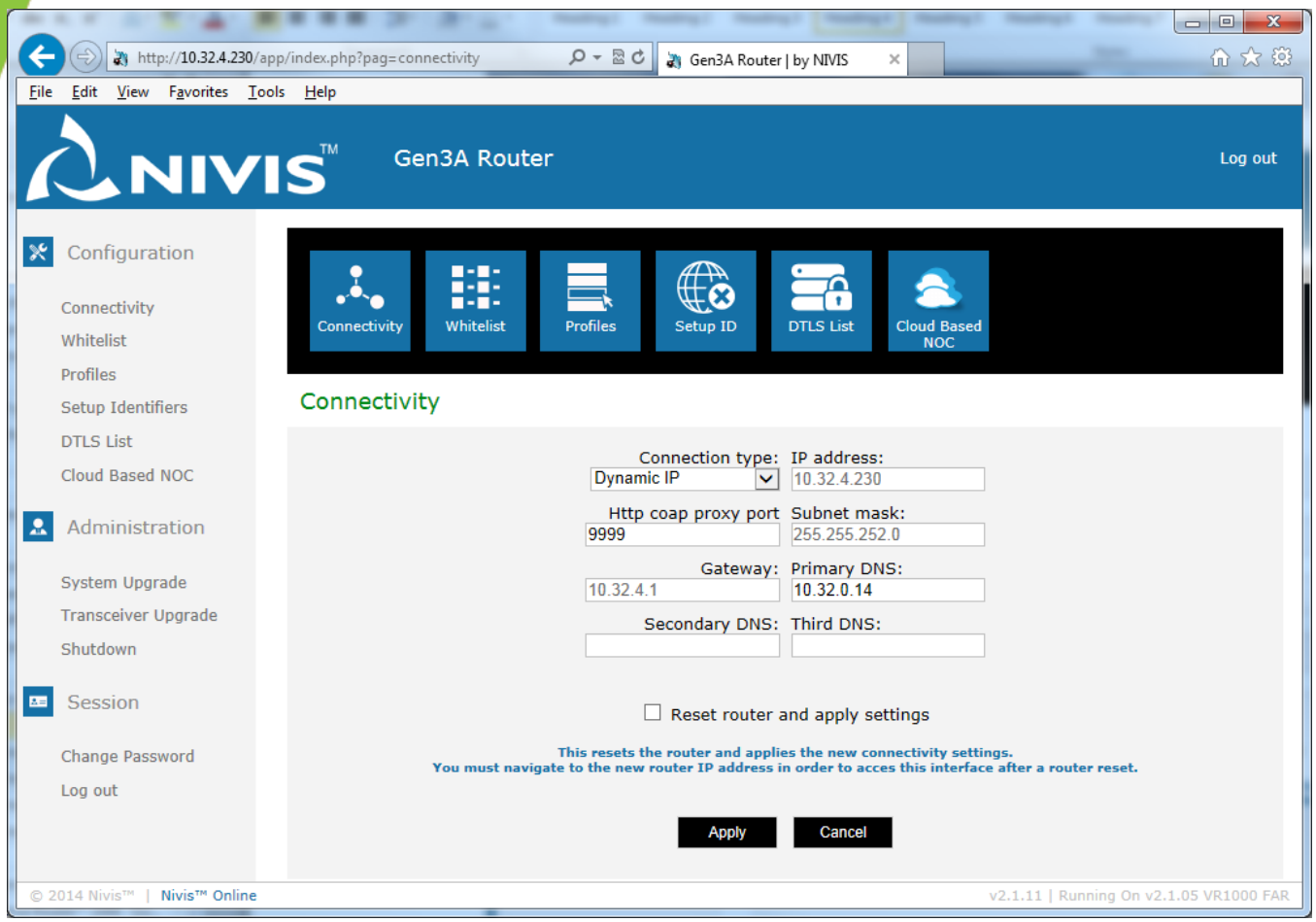
After login, the website shows the configuration/connectivity page.

All of the website pages are accessible through the left-hand menu.

All of the pages of the current section are available through the metro-style menu at top of the page.

5.1.2.2.2 Configuring the Versa Router™ 1000 Quark Connectivity settings

Click on **Connectivity** in the **Configuration** section.



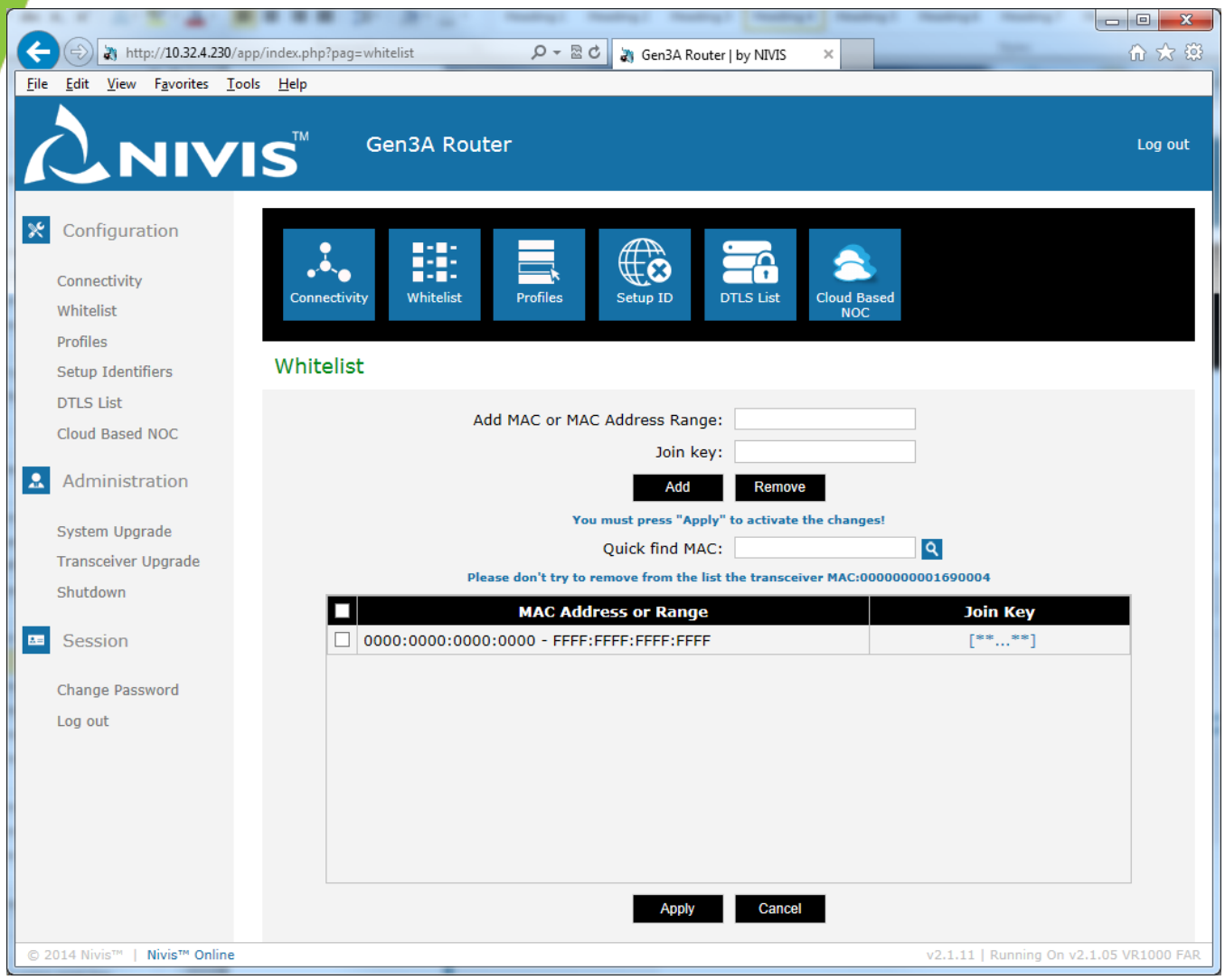
This allows the user to choose between statically allocated IPv4 and dynamically allocated IPv4 (by DHCP) for the Versa Router™ 1000 Quark; set the static IPv4, network mask, gateway, and DNS; and change the HTTP/COAP Proxy port.

WARNING: This page is for advanced users only – do not use it unless you know precisely how to configure the network. Any invalid values may render the router dysfunctional, or may cause difficult-to-trace malfunctions.

NOTE Make sure you are not causing IP conflicts when you make changes.

5.1.2.2.3 Configuring the Versa Router™ 1000 Quark Whitelist

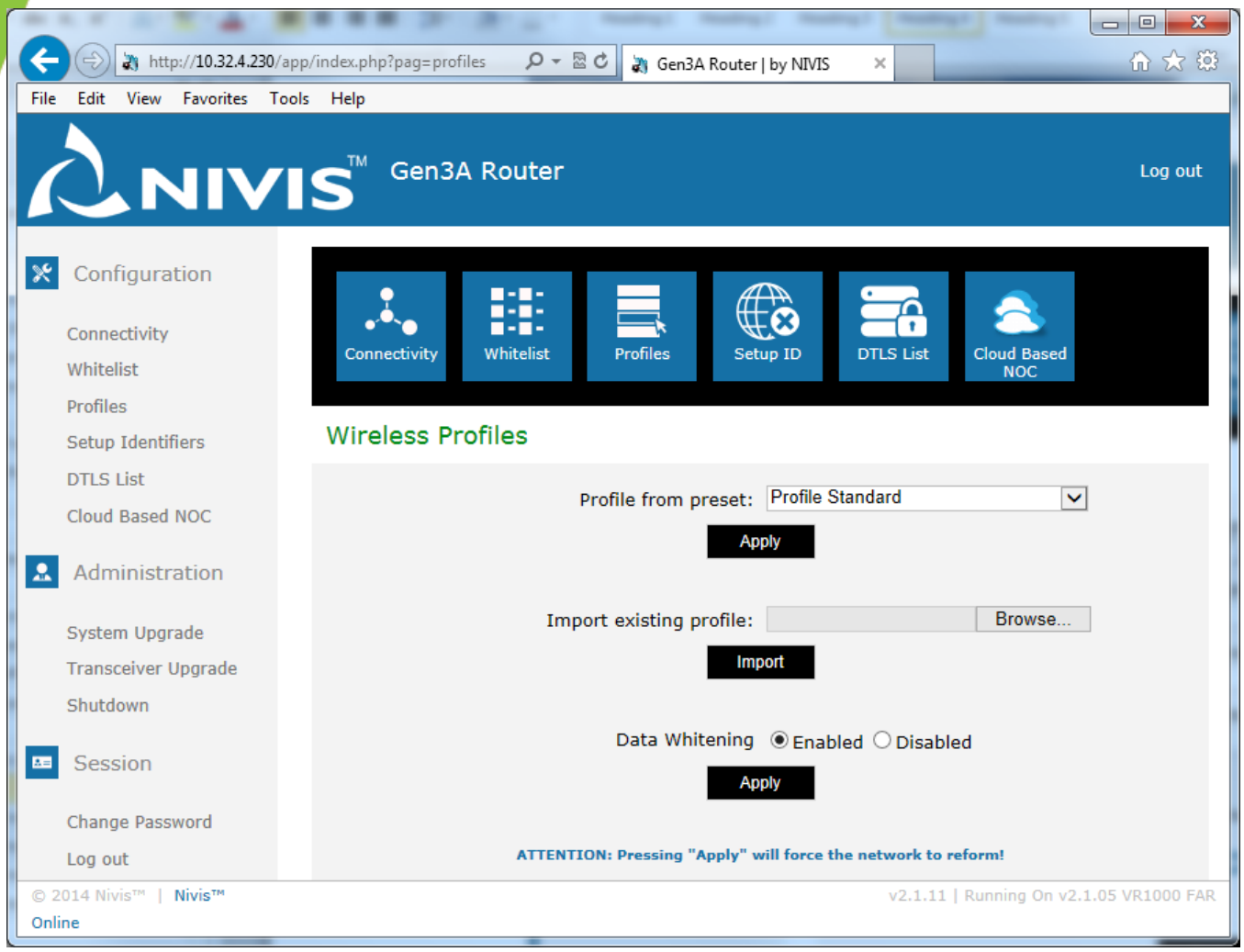
Click on **Whitelist** in the **Configuration** section.



The whitelist is the list with devices accepted in the network and their respective join keys. Devices are specified by their MAC addresses. There are two ways to specify the whitelist: by specifying each MAC individually or by using a MAC range.

5.1.2.2.4 Configuring the Versa Router™ 1000 Quark Profiles

Click on **Profiles** in the **Configuration** section.



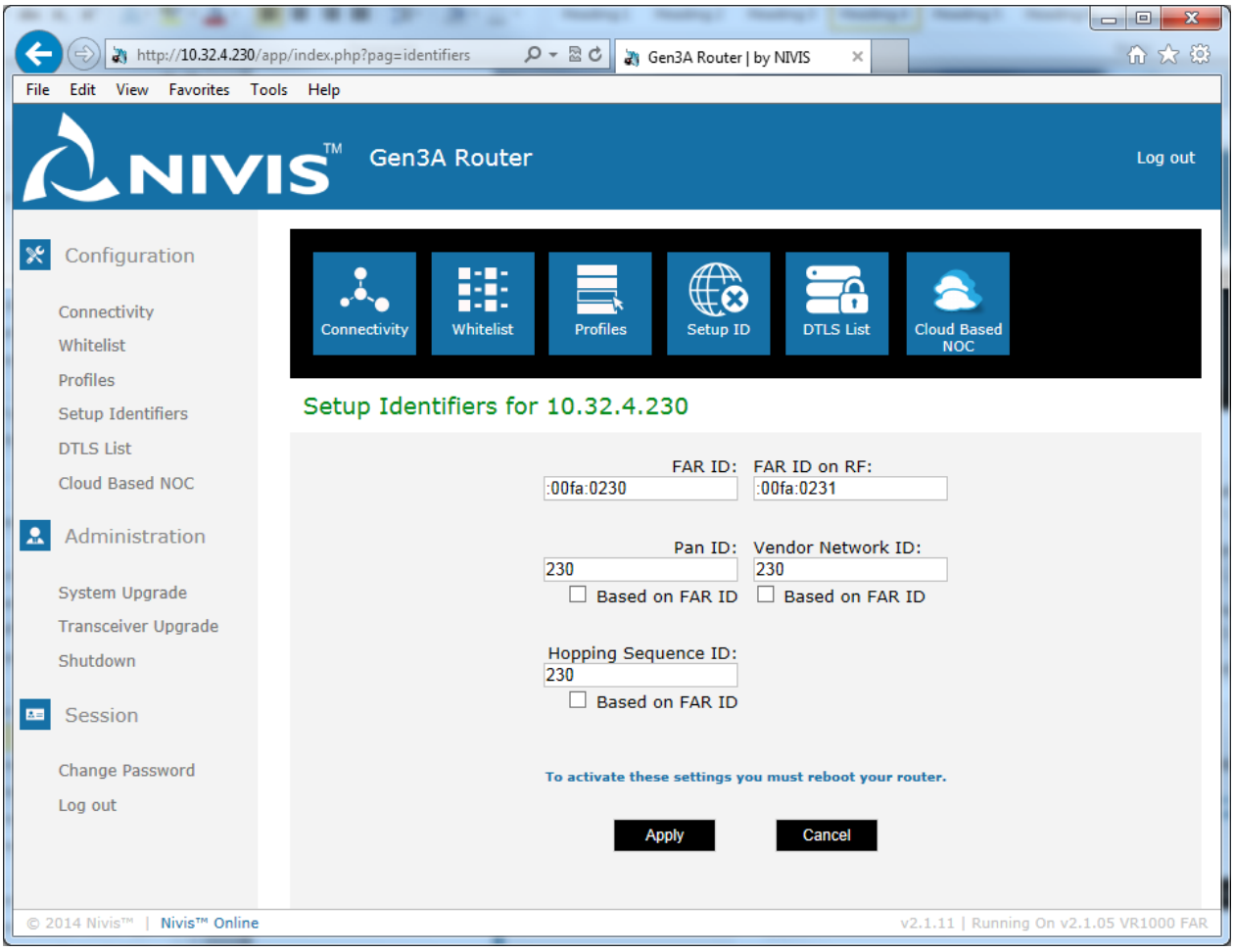
This allows switching between “Demo” setup (small network: up to 10 devices, fast join), “Standard” setup (average network size: up to 100 devices, average join), and “Large” setup (large network size, up to 500 devices). This also enables users to load a custom profile file.

NOTE Use only Nivis-provided profile files. Do not edit Nivis-provided profile files.

On this page, the user can also enable/disable the “Data whitening” feature.

5.1.2.2.5 Configuring the Versa Router™ 1000 Quark Identifiers

Click on **Setup Identifiers** in the **Configuration** section.



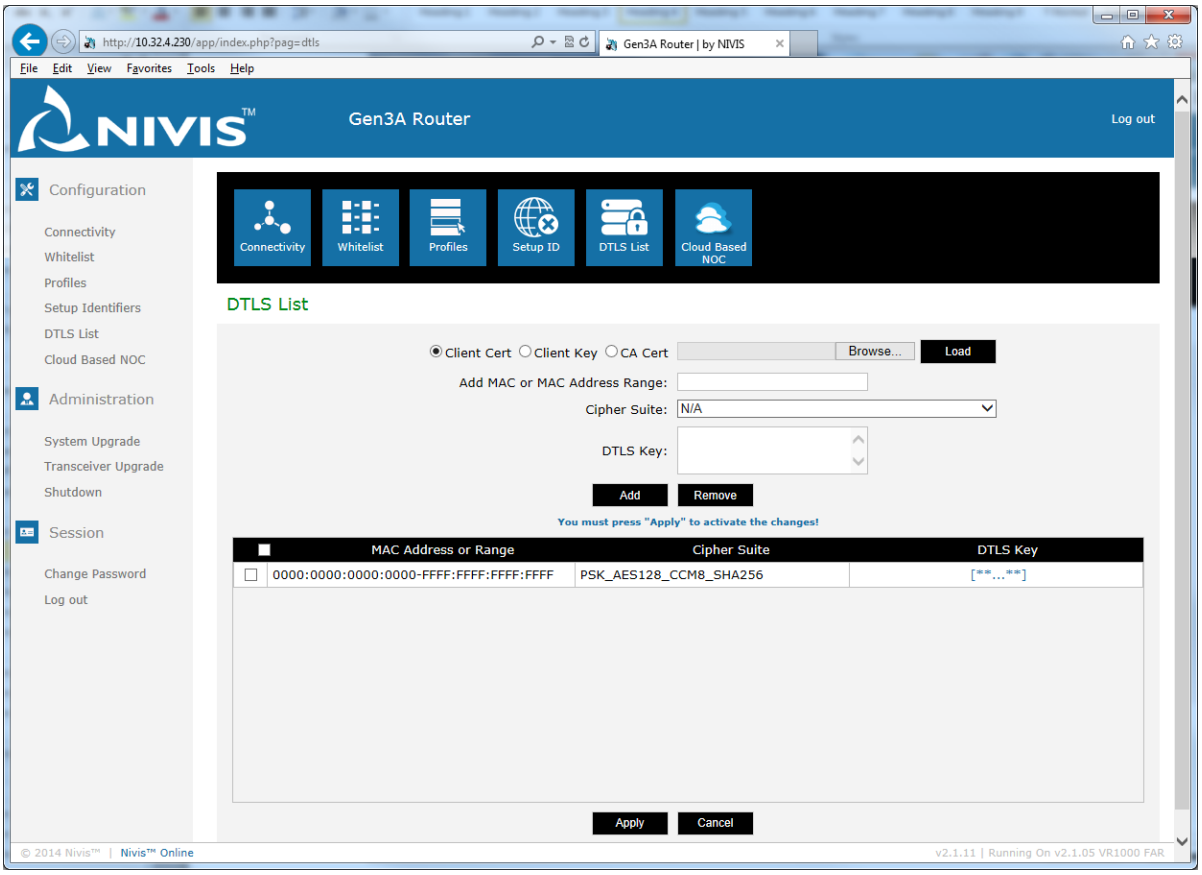
The default setup identifiers need to be changed **only** if you have several collocated networks/Quarks. In this case, each collocated Quark must have its own unique set of identifiers.

For ease of configuration, use the “Based on FAR ID” checkbox where available.

WARNING Do not change the variables FAR ID and FAR ID on RF under any circumstance.

5.1.2.2.6 Configuring the Versa Router™ 1000 Quark DTLS list

Click on **DTLS List** in the **Configuration** section.



On this page, the user can load certificates and configure the DTLS list.

User actions:

- **Load**: The user will select one of the three radio buttons (ClientCert, ClientKey, or CA Cert), will choose a local .pem file, and then will load the certificate on the router.
- **Add**: The Mac, Cipher Suit, and DTLS Key fields will be validated and a new entry will be added to the table.
- **Remove**: The rows with the checkbox checked will be removed.
- **Apply**: The changes will be saved on the router.
- **Cancel**: The pending changes will be removed and the table will be reloaded from the router.
- **Select from table** (by clicking in the checkbox or on the row): The selected MAC Address or Range will be loaded in the textbox from above; the corresponding Cipher Suite will be loaded in the drop down list from above; and the DTLS key will not be loaded. Also, the selected device details will be bolded in the table.
- **Update**: If the user will edit the loaded data (or input a new key), the Add button will be renamed Update and will trigger the update in the table.

Keys will be used only if the Cipher Suite is one of the following:

- PSK_NULL_SHA256
- PSK_AES128_CCM_SHA256
- PSK_AES128_CCM8_SHA256

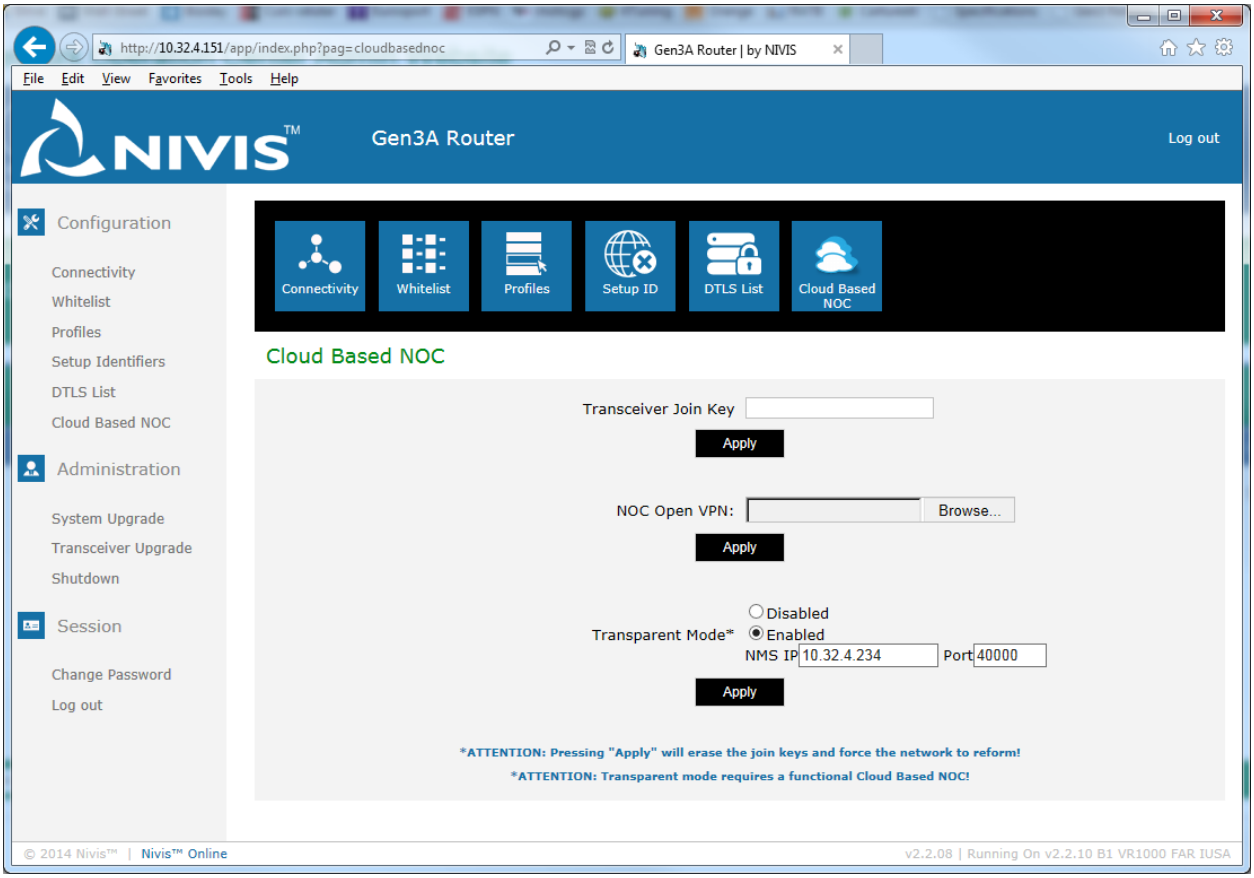
The Cipher Suite “ECDHE_ECDSA_AES128_CCM8_SHA256”uses a Certificate, not a Key.

The Cipher Suite “N/A” does not use a Key or Certificate.

5.1.2.2.7 *Configuring the Versa Router™ 1000 Quark Transparent Mode - connection to Cloud-Based NOC*

- The Quark can be configured through Quark website to work in two different modes:
  - Standalone gateway
  - Router connected to the NMS (transparent mode enabled)
- When operating as a standalone gateway, the Quark is offering the following functionalities:
  - Resource directory
  - HTTP-CoAP Proxy
  - Data caching and storage
  - Support for nodes firmware upgrade
  - Support for NAMT for network monitoring, management, configuration, topology view, etc
- When operating connected to Nivis NMS the functionalities above move to the NMS and are no longer available on the Quark
- Main common functionalities:
  - Border router for 6lowpan/RPL network, WPAN coordinator
  - IP Routing
  - Name service (DNS)

Click on "Cloud Based NOC" link in "Configuration" section. This link is available only for routers with "Cloud Based NOC" capability.



User actions available on this page:

- Set a new join key for the transceiver. This set the join key on the Transceiver itself, NOT the corresponding entry in the Quark Whitelist.
- Upload NOC Open VPN certificates.
- Set transparent mode ON (for this the NMS IP and Port are required) or OFF.

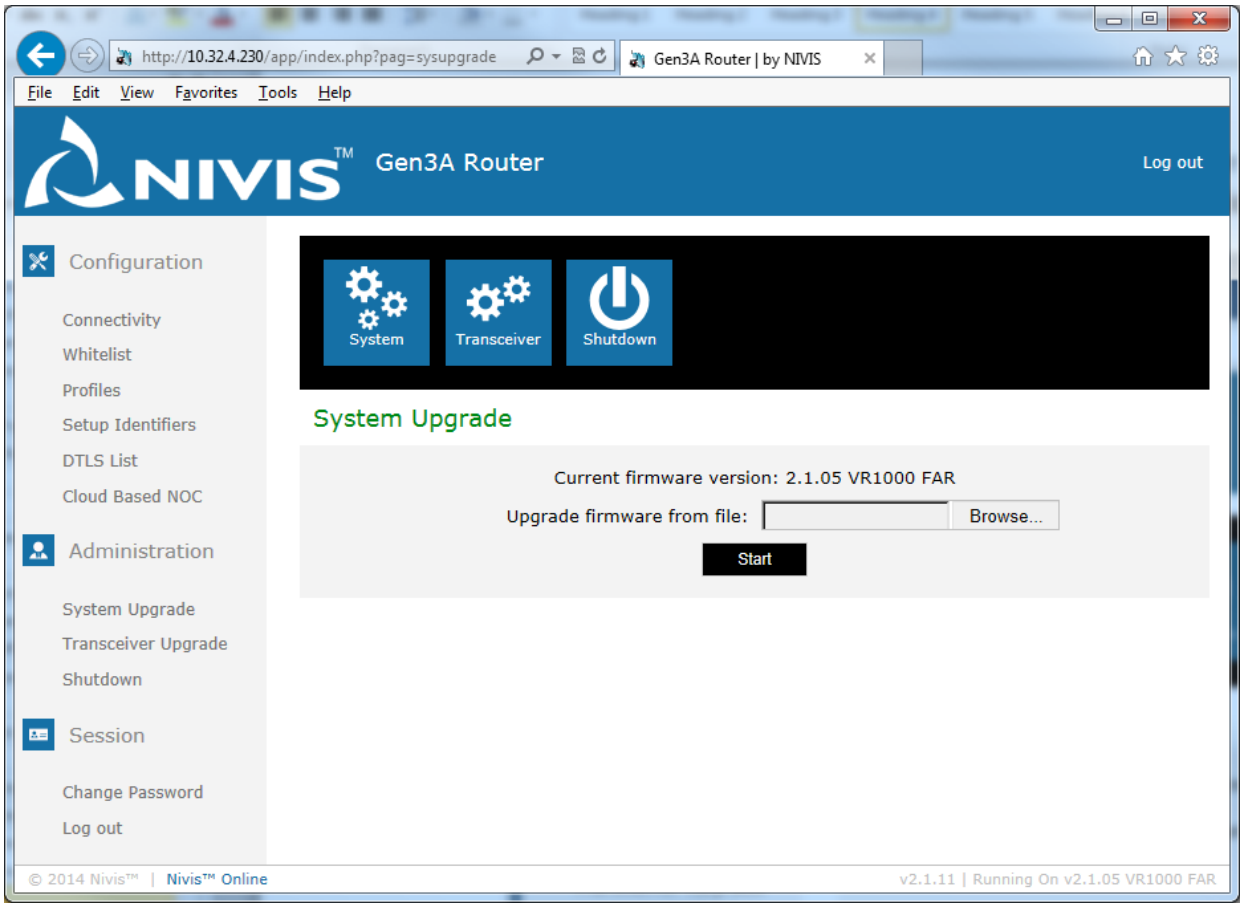
  When transparent mode is ON, the user cannot do the following actions:

  - Modify whitelists in “Whitelists” page.
  - Modify DTLS lists in “DTLS List” page.
  - Load certificates in “DTLS List” page.

WARNING The transceiver join key must be set ONLY to the default join key set at Production. Using any other value for Transceiver join key will result in transceiver and the whole associated network not joining to the system

5.1.2.2.8 Upgrading the Versa Router™ 1000 Quark Software

Click on **System Upgrade** in the **Administration** section.



If you receive a new Quark software archive file, you can activate it using this page.

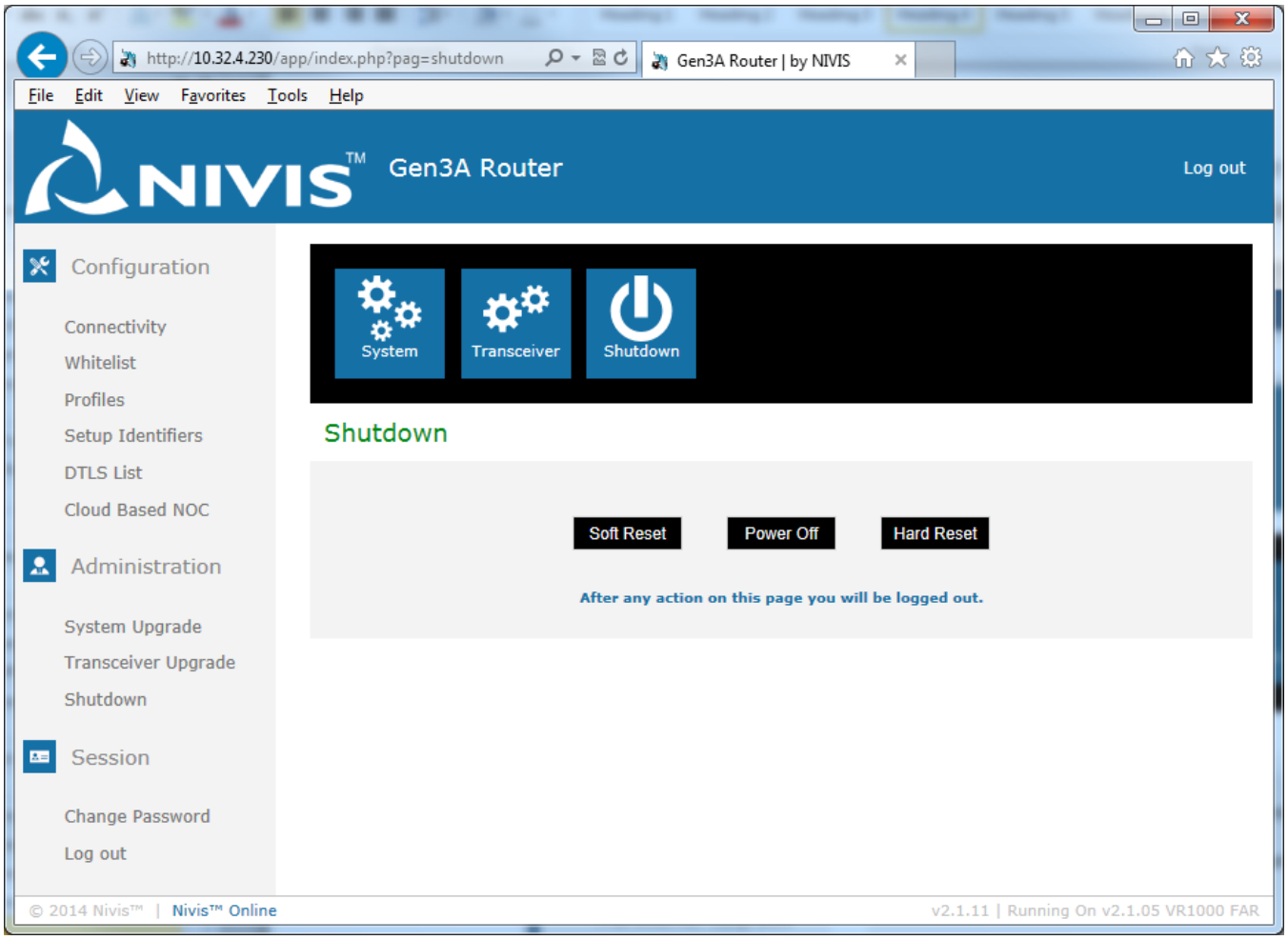
Browse to the Nivis-provided software archive file, then click **Start**.

The upgrade process takes several minutes. At the end of the process, the board will restart, rendering the Quark and the Quark website unavailable for up to several minutes.

5.1.2.2.9 Upgrading the Versa Router™ 1000 Quark Transceiver

The radio on the Versa Router™ 1000 Quark (the Transceiver) is upgraded separately from the main Quark software.

Click on **Transceiver Upgrade** in the **Administration** section.



If you receive a new Transceiver firmware file, you can activate it using this page.

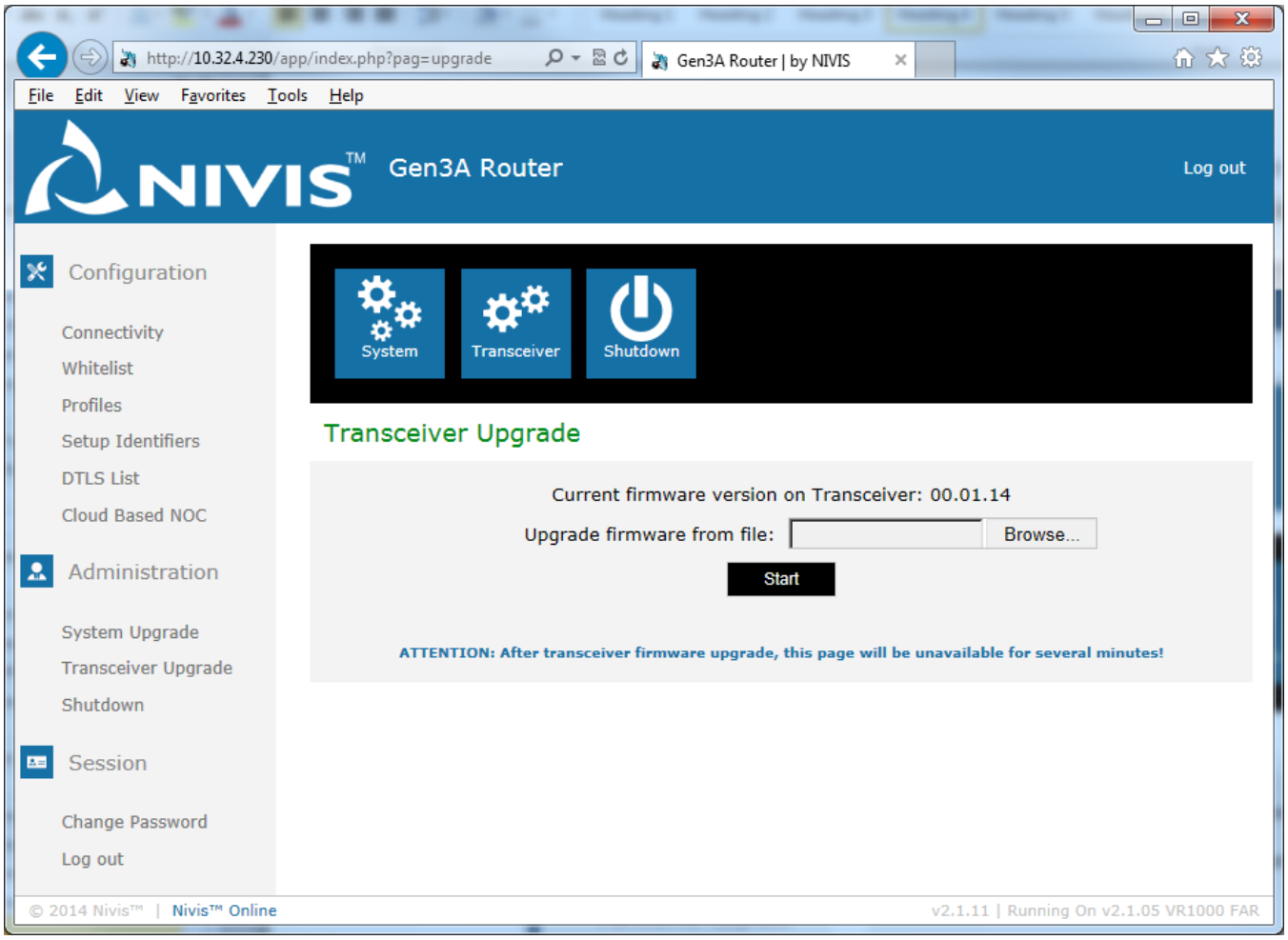
Browse to the Nivis-provided firmware file, then click **Start**.

The upgrade process takes up to a minute. At the end of the process, the Transceiver will restart, beginning a full rejoin of the network.

5.1.2.2.10 Shutting down the Versa Router™ 1000 Quark

The administration page allows the user to perform a soft reset (restarting the apps), hard reset (rebooting the board),or power off procedure (graceful shutdown).

Click on **Shutdown** in the **Administration** section, then choose the desired shutdown type.

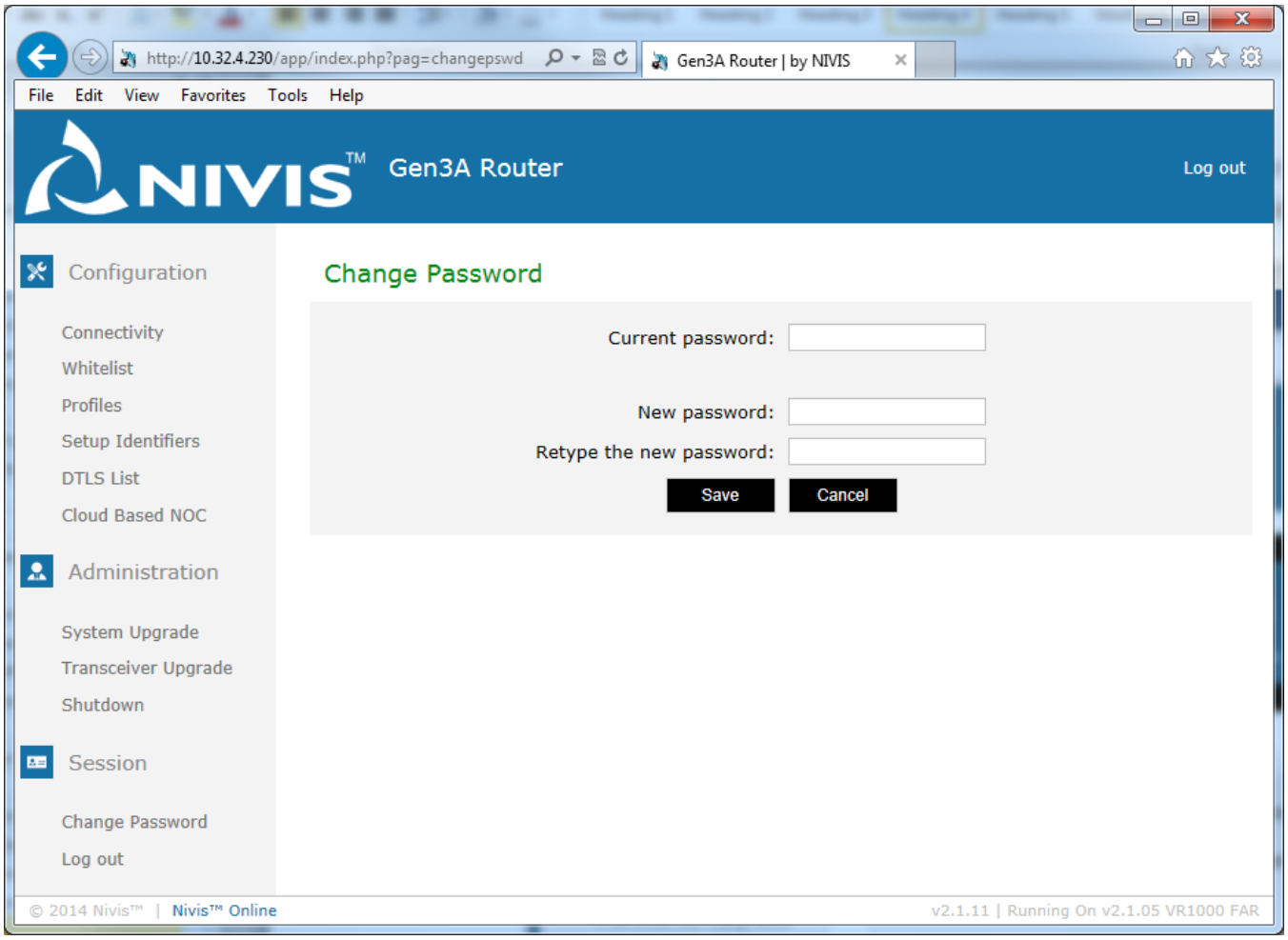


WARNING Before removing the Quark power, you **MUST** perform a graceful shutdown (click on **Power Off**). Failure to do so may render the Versa Router™ 1000 Quark non-functional.

5.1.2.2.11 Changing the Versa Router™ 1000 Quark Website password

The administration page allows users to change the Versa Router™ 1000 Quark website password.

Click on **Change Password** in the **Session** section. Type the old password and new password, re-type the new password, and click **Save**.



5.1.2.2.12 Logging out

Click on **Log Out** in the **Session** section to log out of the website.

5.1.2.3 Web-based Administration /admin/

The web-based administration /admin/ interface is the secondary method that can be used to administer/configure the Versa Router™ 1000 Quark. It requires a web browser and the IP of the Versa Router™ 1000 Quark. The Versa Router™ 1000 Quark must be connected to the local LAN and then powered on, and the IP/mask of the router must be accessible from the PC where the browser is running.

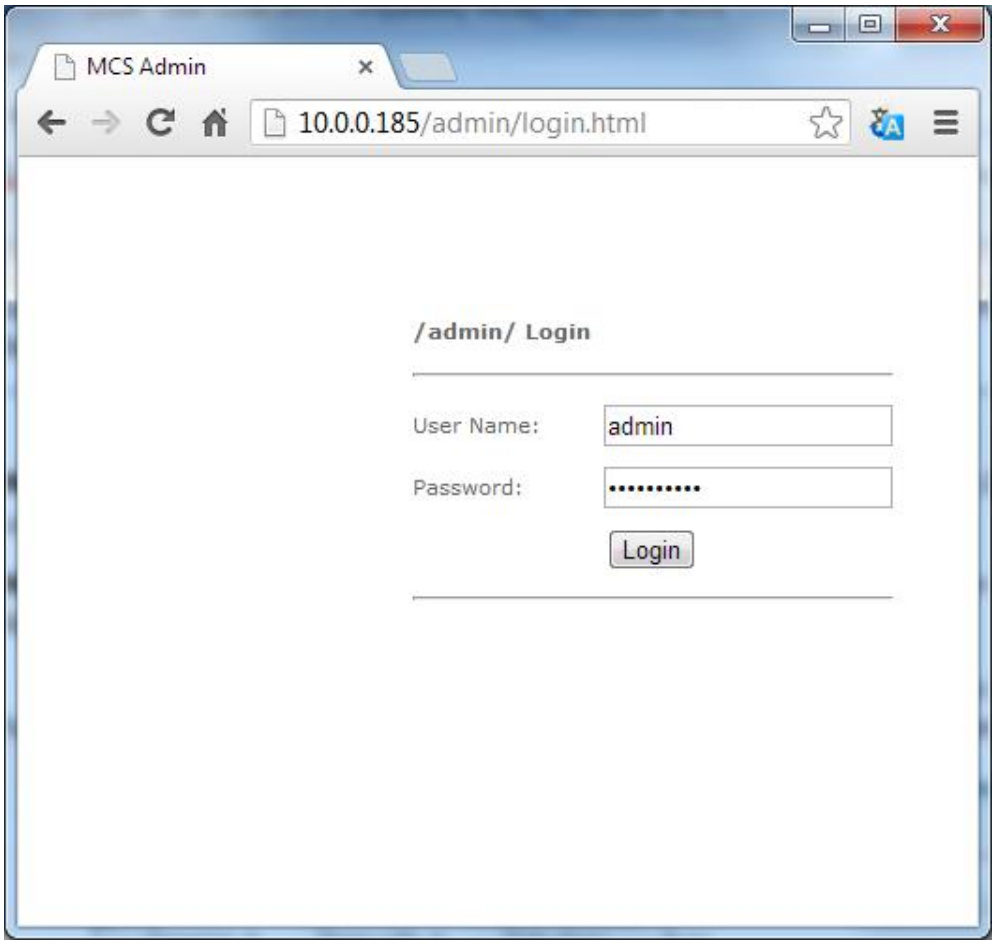
NOTES

The /admin/ interface is for advanced users only. Do not use this method unless instructed by a Nivis representative.

Depending on the firmware running on the Versa Router™ 1000 Quark, more or fewer features may be available than described herein.

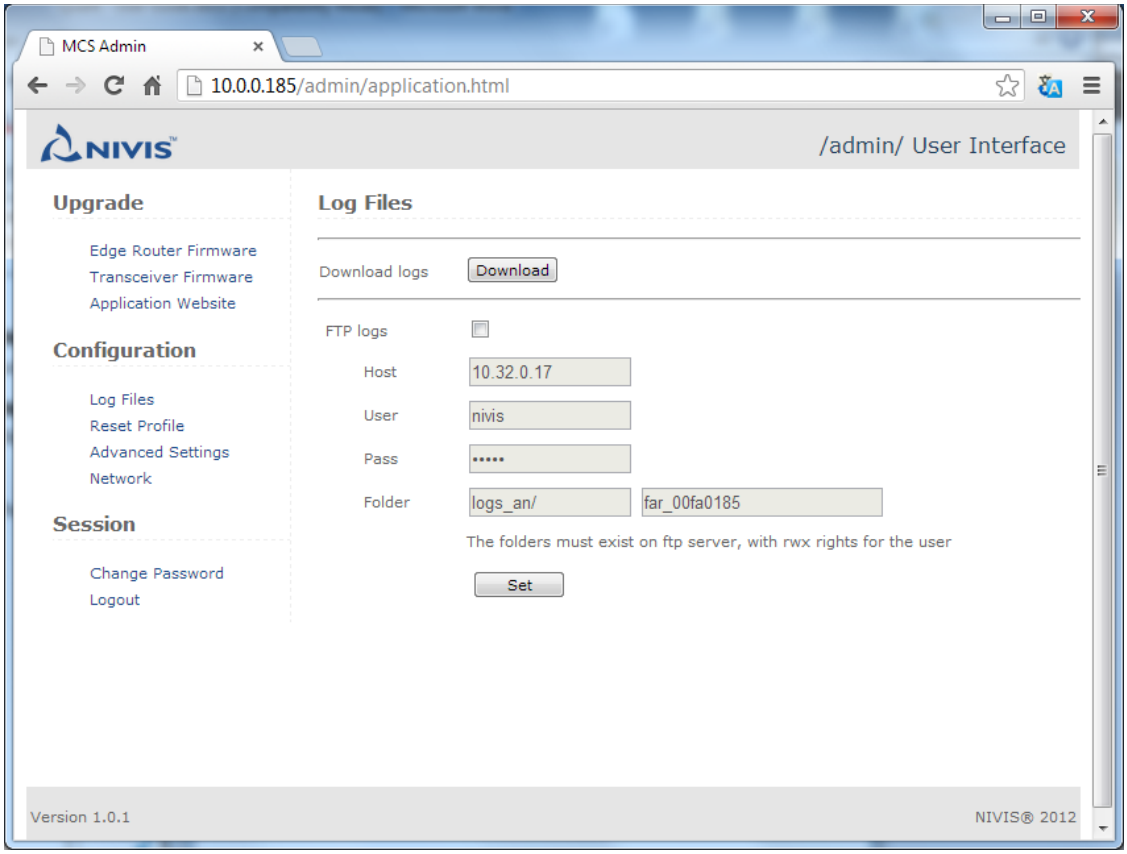
All web administration tasks require login. The login step is presented here only once. It is understood that all of the following steps will require it.

1. Navigate to the following URL: http://<ER_IP>/admin where<ER_IP> is replaced by the Versa Router™ 1000 Quark IP.



2. Type the following credentials in the input fields
   - User: **admin**
   - Password: **adminadmin**
3. Click **Login**.

The following page appears, allowing access to various tasks. By default, the **Log Files** link is selected.



NOTE

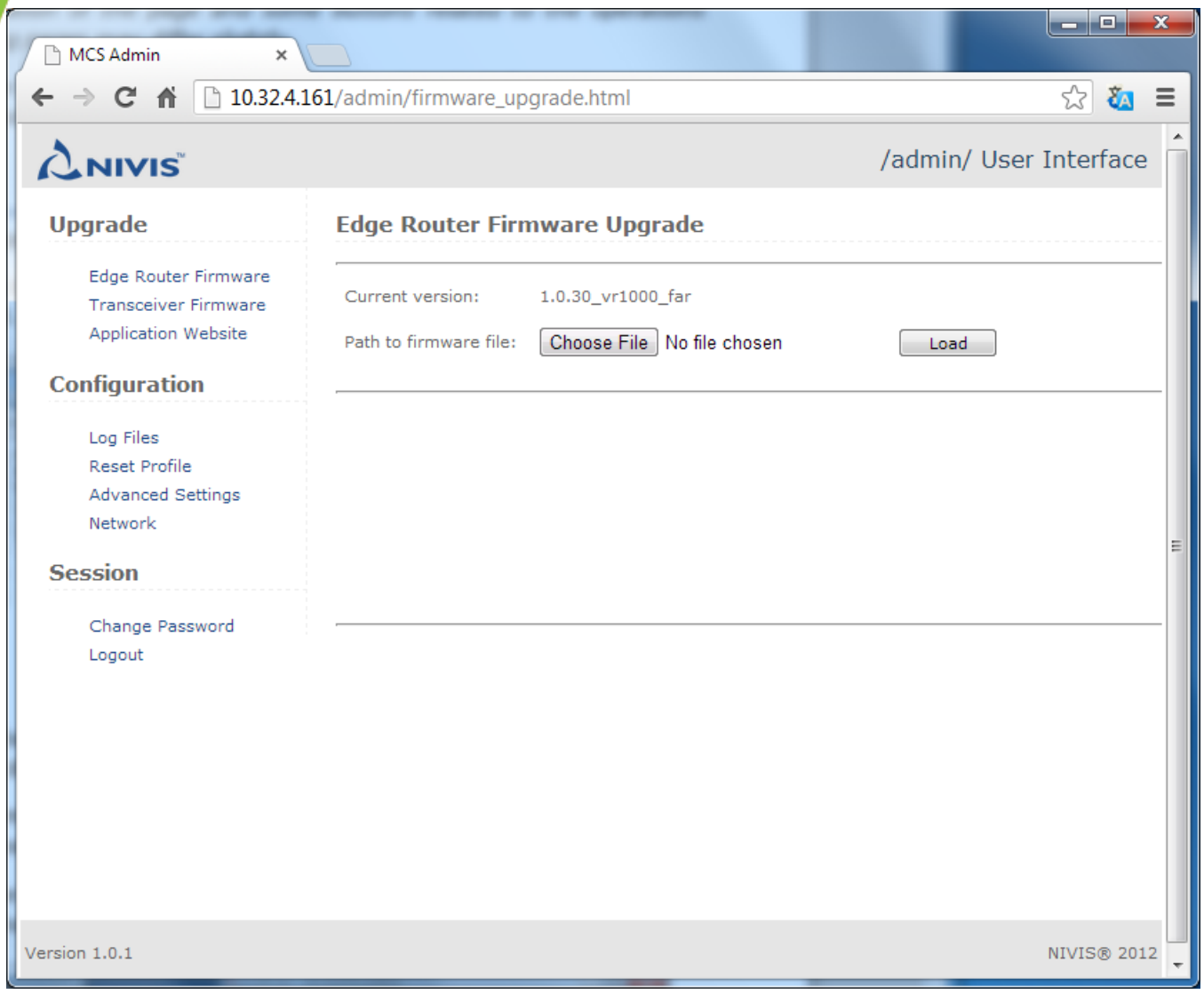
Depending on the web browser you are using to log into the Versa Router™ 1000 Quark Administration, the graphical representation of the page and some buttons related to the operations described in the following pages may differ slightly.

For instance, the button **Choose File** in Google Chrome and Apple Safari is the same as the button **Browse** in Mozilla Firefox and Internet Explorer.

5.1.2.3.1 Upgrade Versa Router™ 1000 Quark Software

This function is used to upgrade the Quark SW when an update is provided by Nivis.

1. Click **Edge Router Firmware**. The following screen shows the version currently installed.



2. Click **Browse** to locate and select a new firmware file, then click **Load** (the version must be different from the previous one).
3. Wait until the firmware is activated. This process will take few minutes. Do not power-cycle the board or interrupt the upgrade processing any other way.

During the upgrade, the process log is displayed. When the operation is complete, the screen should look similar to the screen capture below.

Operation complete.
Log results:

```
+ ./web_upgrade.sh
Stopping modules
Announce to stop modules: watchdog.sh
killall watchdog.sh
Module watchdog.sh ended OK
All modules ended OK
Announce to stop modules: RplBridge RplRoot RemoteAccess SNMP_Agent DNS_Server NMS_EntryPoint HttpCoapPr
killall RplBridge RplRoot RemoteAccess SNMP_Agent DNS_Server NMS_EntryPoint HttpCoapProxy sys_monitor.sh
Module RplBridge ended OK
Module RplRoot ended OK
Module RemoteAccess ended OK
  625 ?        S1     7:06 ./SNMP_Agent
29997 ?        S      0:00 ./SNMP_Agent
Module DNS_Server ended OK
17644 ?        S1     0:51 ./NMS_EntryPoint
29999 ?        S      0:00 ./NMS_EntryPoint
Module HttpCoapProxy ended OK
Module sys_monitor.sh ended OK
Module log_watcher.sh ended OK
Module SNMP_Agent ended OK
17644 ?        S      0:51 ./NMS_EntryPoint
29999 ?        S      0:00 ./NMS_EntryPoint
17644 ?        S      0:51 ./NMS_EntryPoint
29999 ?        S      0:00 ./NMS_EntryPoint
Module NMS_EntryPoint ended OK
All modules ended OK
Validate DB '/jffs/nivis/far/tmp/EntryPoint.db3' against '/jffs/nivis/far/activity_files/EntryPoint_Fixt
Move DB to BackupDB /jffs/nivis/far/tmp/EntryPoint.db3 -> /jffs/nivis/far/activity_files/EntryPoint.db3
== STARTING ACT_OUT ===
activate.sh --fw_file /jffs/nivis/far/tmp/upgrade_web/an_bin_1.0.11_b1_p1025twr_far_lg.tgz
SetActive firmware: fw_file=/jffs/nivis/far/tmp/upgrade_web/an_bin_1.0.11_b1_p1025twr_far_lg.tgz fw_ver=
SetActive  [an_bin_1.0.11_b1_p1025twr_far_lg]
SetActive firmware [an_bin_1.0.11_b1_p1025twr_far_lg] done. Restart in 30sec.
ER_RESULT=SUCCESS
```

NOTES

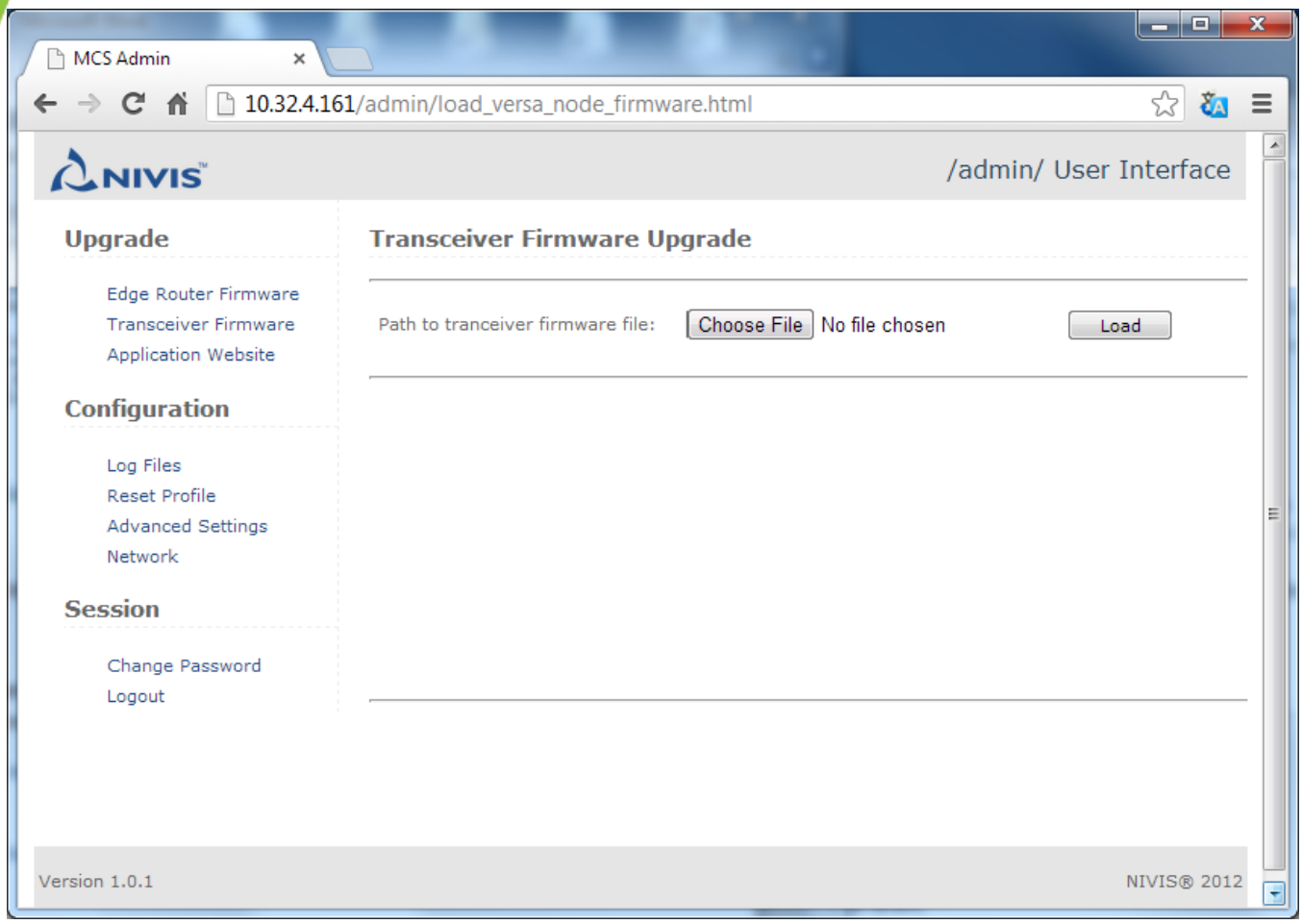
Contact Nivis support if the upgrade operation does not succeed.

Power-cycle all Smart Object devices after performing a Quark firmware upgrade, then wait about 5 minutes before attempting to connect the NAMT to the Versa Router™ 1000 Quark.

5.1.2.3.2 Upgrade Versa Router™ 1000 Quark Transceiver firmware

This function is used to upgrade the Quark transceiver firmware when an update is provided by Nivis.

1. Click **Transceiver Firmware**. The following screen appears.



2. Click **Browse** to locate and select a new firmware file, then click **Load** (the version must be different from the previous one).
3. Wait until the firmware is activated. This process will take few minutes. Do not power-cycle the board or interrupt the upgrade processing any other way.

After the upgrade, the process log is displayed.

NOTES

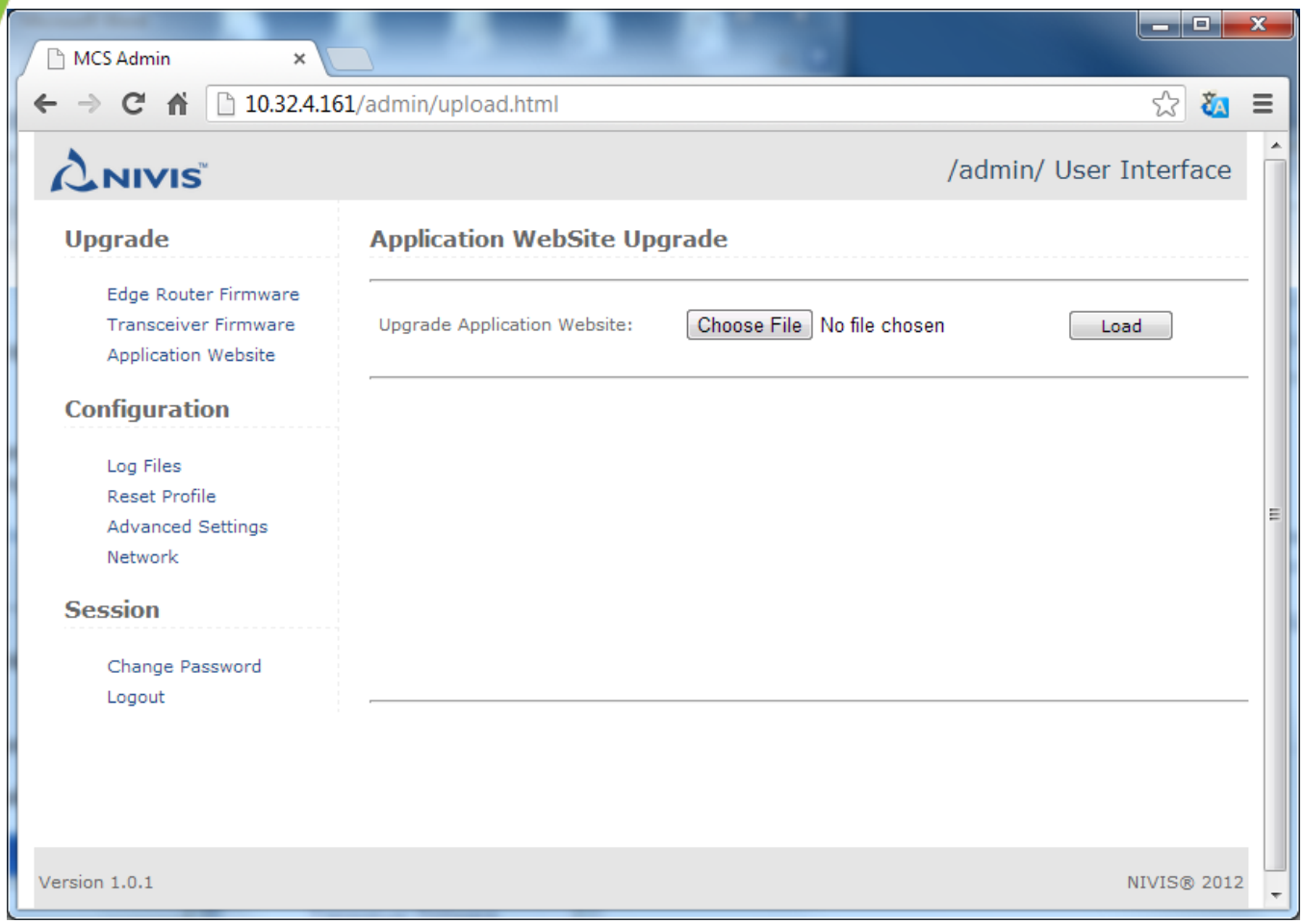
Contact Nivis support if the operation does not succeed.

Power-cycle all Smart Object devices after performing a Quark transceiver firmware upgrade, then wait about 5 minutes before attempting to connect the NAMT to the Versa Router™ 1000 Quark.

5.1.2.3.3 Upgrade the Versa Router™ 1000 Quark Website

This function is used to upgrade the Quark website when an update is provided by Nivis.

1. Click **Application Website**. The following screen appears.



2. Click **Browse** to locate and select a new firmware file, then click **Load** (the version must be different from the previous one).
3. Wait until the website is activated. This process will take few minutes. Do not power-cycle the board or interrupt the upgrade processing any other way.

After the upgrade, the process log is displayed.

NOTES

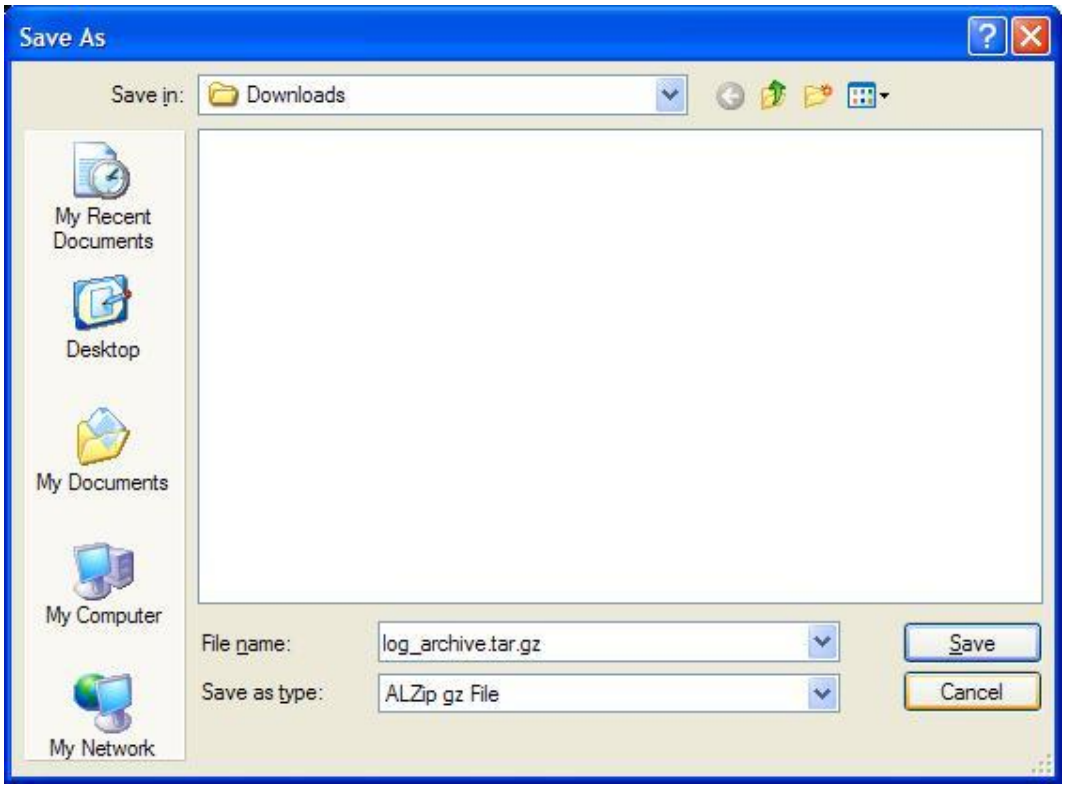
Contact Nivis support if the operation does not succeed.

Power-cycle all Smart Object devices after performing a Quark website upgrade, then wait about 5 minutes before attempting to connect NAMT to the Versa Router™ 1000 Quark.

5.1.2.3.4 Download Versa Router™ 1000 Quark Logs

This function provides the Versa Router™ 1000 Quark logs when a log snapshot is requested for troubleshooting purposes.

1. Click **Download Logs**. A window opens prompting you to open or save the archive.
2. Click **Save**. The **Save As** dialog will open.



3. In this dialog, choose the location for the archive and click **Save**.

5.1.2.3.5 Set the Versa Router™ 1000 Quark for FTP logs upload

This page allows you to configure the Versa Router™ 1000 Quark to upload the logs to a FTP server when long-term logs are needed and the log snapshot does not provide enough information.

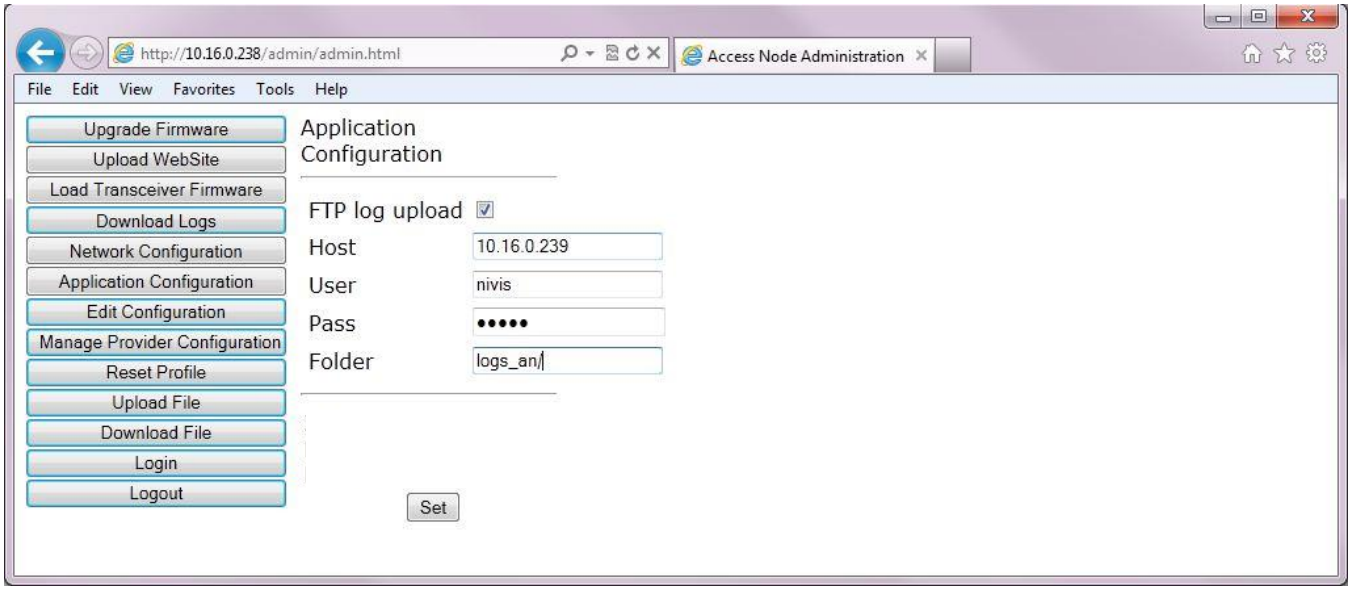
WARNING This page is for advanced users only – do not use unless you have been instructed by a Nivis representative.

Because of space restrictions, the logs on the Versa Router™ 1000 Quark are frequently removed. In order to retain logs over a longer period of time, a FTP server can be used. The Versa Router™ 1000 Quark must be configured to move the logs onto the FTP server instead of removing them.

The FTP server must meet the following conditions:

1. Be in the same network with the Versa Router™ 1000 Quark. It **must be available in the network all the time**; otherwise, the VR may not function correctly.
2. Be UNIX-compatible.
3. Have a username and password created. Anonymous users should not be used.

NOTE Please be aware that the most recent logs will still be on the Versa Router™ 1000 Quark (available through log snapshot: **Download Versa Router™ 1000 Quark Logs** button) and not on the FTP server.



To configure the Versa Router™ 1000 Quark to upload the logs to a FTP server, on the Application Configuration screen:

1. Select the **FTP log upload** checkbox.
2. In the **Host** field, enter a valid FTP server IP address.
3. In the **User** and **Pass** input fields, enter a valid username and password for the FTP server. Do not use anonymous users.
4. Select the folder on the FTP server in which the logs are to be saved.

5. Click **Set**.

5.1.2.3.6 Reset profile

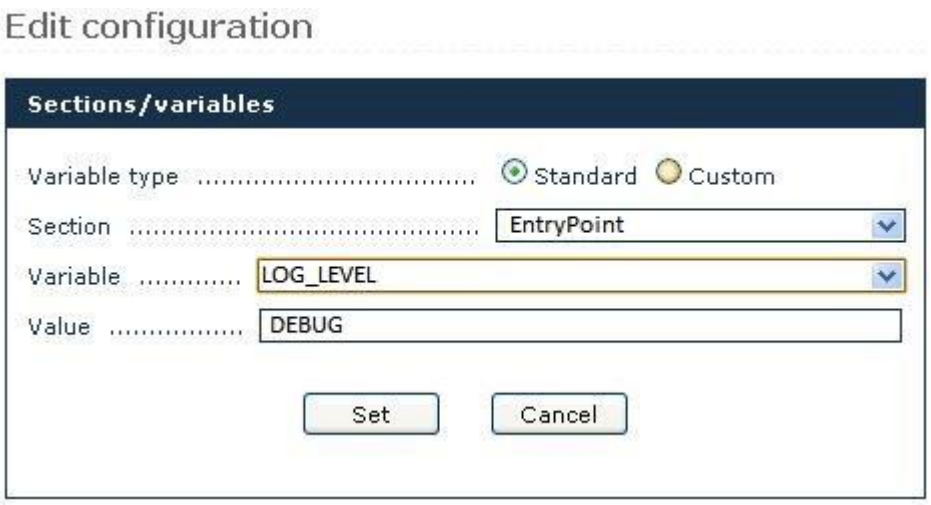
WARNING The **Reset Profile** section is for troubleshooting purposes only. **Do not use** unless instructed specifically by a Nivis representative.

5.1.2.3.7 Edit Versa Router™ 1000 Quark General Configuration

This page allows you to view/set less common configuration variables.

WARNING This page is for advanced users only – do not use unless you have been instructed specifically by a Nivis representative on what values to change. Incorrect values may render the router non-functional or may cause difficult-to-trace malfunctions.

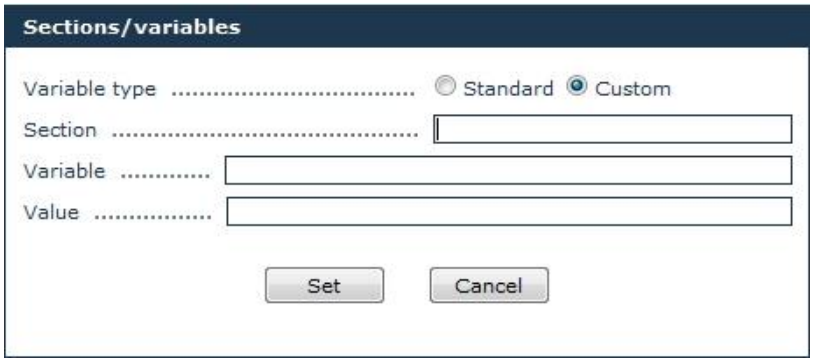
1. Click on **Advanced Settings**. The following form will open to the right of the operation list:



2. In the form, select a **Section** in the drop-down list. The **Variable** list will change accordingly.
3. Select a **Variable** in the drop-down list.
4. Set /edit the **Value** field, then click **Set**.

WARNING Do not change [GLOBAL].AN_ID under any circumstance.

To add a new variable, select **Custom** under **Variable type**. The **Section/Variable** items will be empty.



Type the desired information in the **Section**, **Variable**, and **Value** fields and click **Set**.

5.1.2.3.8 Edit Versa Router™ 1000 Quark IP and other Network Settings

This page allows you to view/set network-related settings: IP address/mask/gateway, name servers used, time servers used.

WARNING This page is for advanced users only – do not use it unless you know precisely how to configure the network. Any invalid values may render the router dysfunctional, or may cause difficult to trace malfunctions.

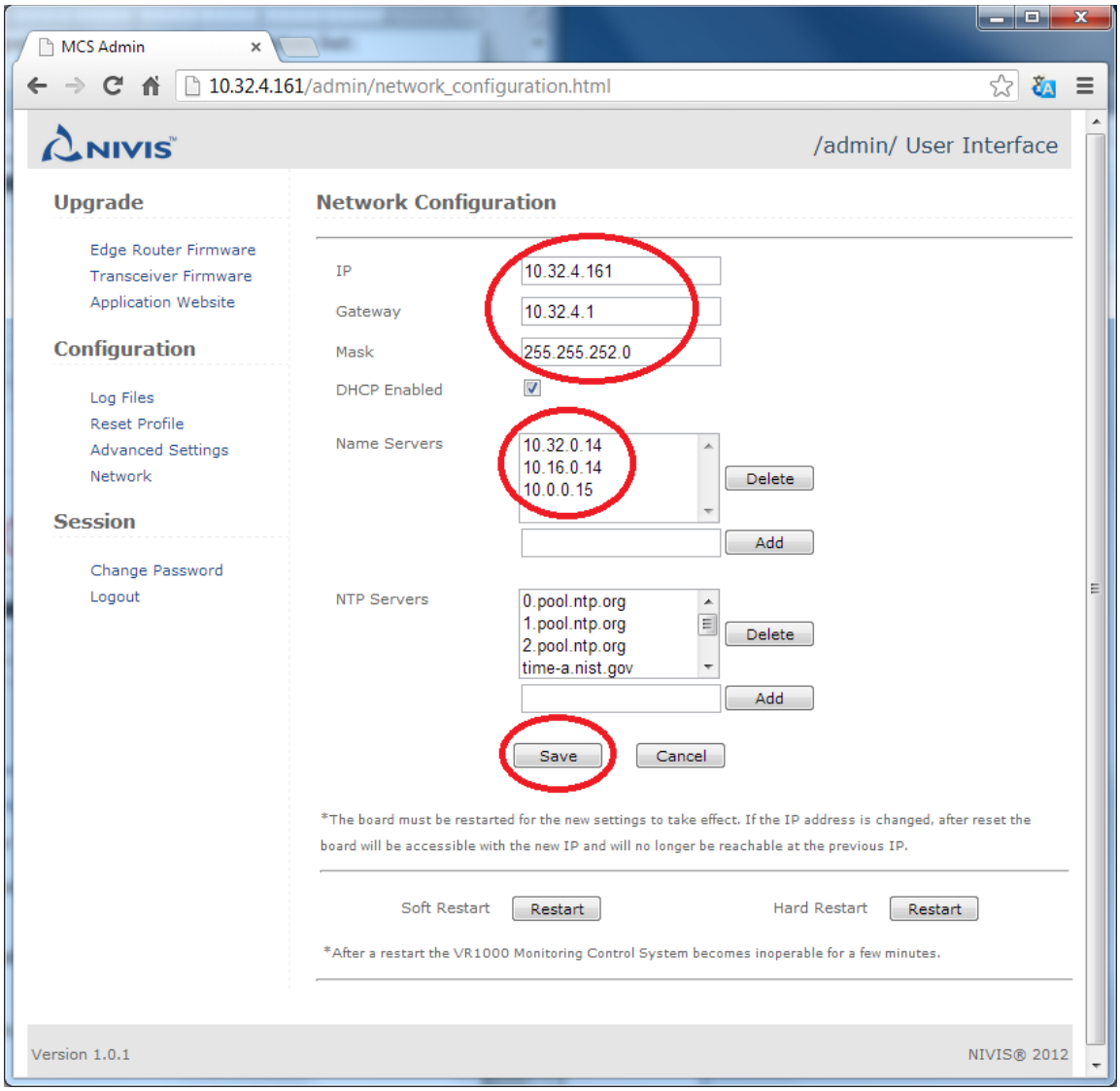
NOTES

Make sure you are not causing IP conflicts when you make changes.

Make sure the name servers used are functional and accessible from the Versa Router™ 1000 Quark.

Make sure the NTP servers are functional and accessible from the Versa Router™ 1000 Quark.

1. Click on **Network**. The following form will open to the right of the operation list.



2. In the form, you can edit the input fields for IP, Gateway, and Mask.
3. To add a name server or an NTP server, provide the correct value in the **Add** input field and click **Add**.
4. To delete a name server or an NTP server, select it in the list of existing items and click **Delete**.
5. When you are done, click **Save** to save the settings. The Versa Router™ 1000 Quark will automatically restart.

Clicking the **Soft Restart** button will restart all the application on the Versa Router™ 1000 Quark.

Clicking the **Hard Restart** button will reboot the Versa Router™ 1000 Quark.

5.1.2.4 Ports & interfaces

The following interfaces are usable on the Versa Router™ 1000 Quark:

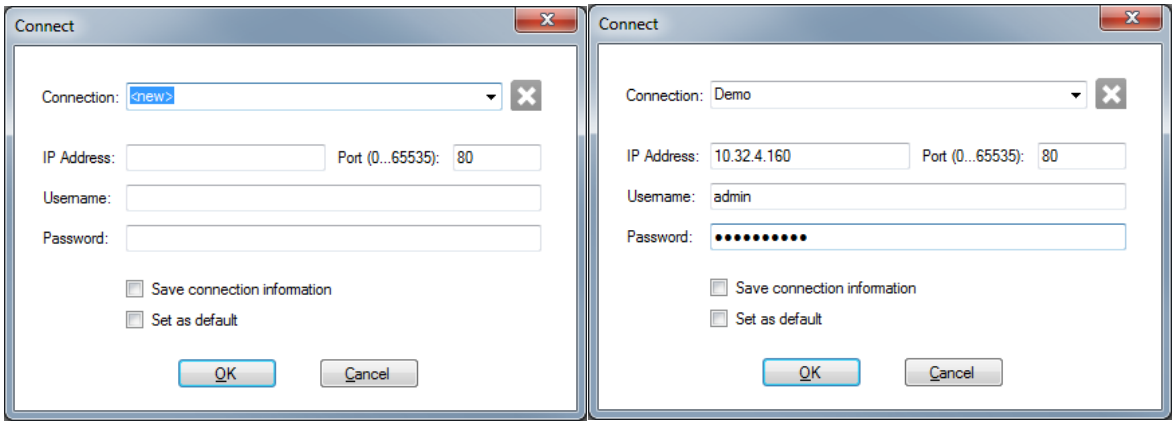
- The **serial port** is used as a kernel console and emergency backup.
- The Versa Router™ 1000 Quark accepts **ssh 22/TCP** connections.
- The Versa Router™ 1000 Quark has an **http** server listening on port **80/TCP** for a Nivis-specific user Interface.
- The Versa Router™ 1000 Quark has an SNMP Agent listening on port **161/UDP**.
- The HTTP – COAP proxy listens on port **9999/TCP** for HTTP queries.
- The Versa Router™ 1000 Quark utilizes the NTP protocol on port **123/USP** to synchronize time with Internet time servers. In order to synchronize the time with Internet time servers, port **123/UDP** must be open in both directions to allow time synchronization.

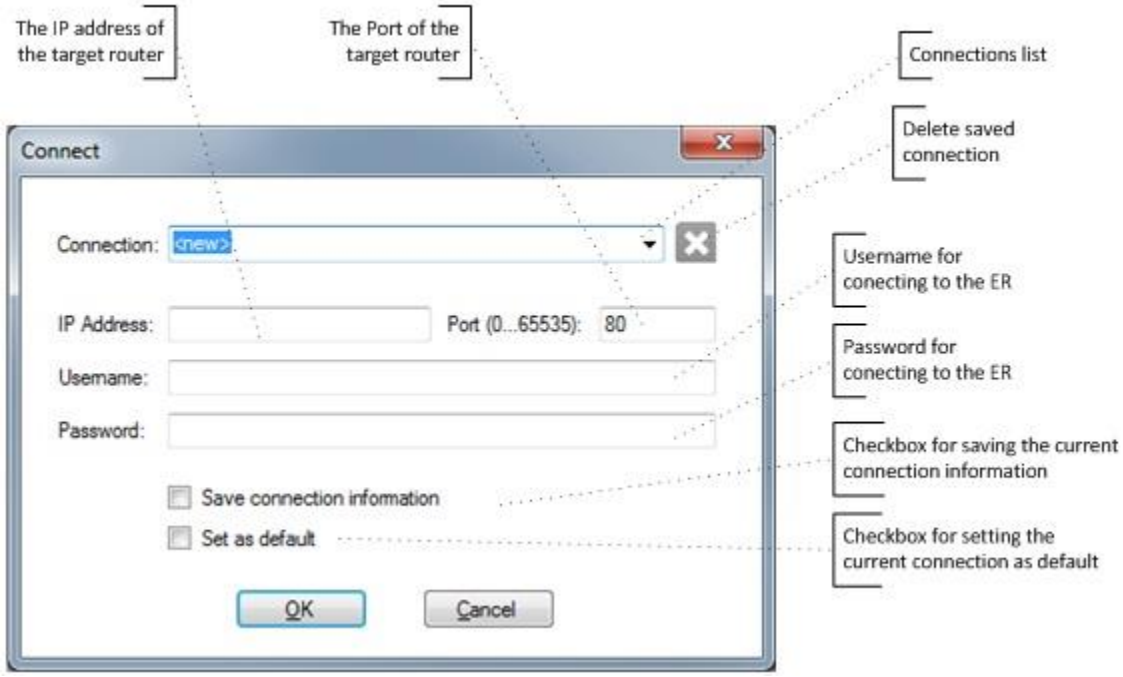
Not all interfaces are guaranteed to be enabled in all cases. Some may be disabled for specific applications.

5.1.3 The NAMT

5.1.3.1 Connecting to a Versa Router™ 1000 Quark

At startup, the NAMT application will display a connection window that allows you to connect to a Versa Router™ 1000 Quark by specifying the connection parameters or by selecting from a list of previously saved connections.





5.1.3.2 Creating a new connection

By default, the connection window will display (read-only) the information for the connection that was saved with the **Set as default** flag set. If no such connection exists, the **Connection** drop-down list will be set on the first position, labeled **<new>**.

If you need to create a new connection, make sure that **<new>** is selected in the connection list. Enter the required connection information (IP address of the ER, username, and password), then click **OK** to connect. Optionally, you can save the connection information by checking the **Save connection information** checkbox and specifying a name for the saved connection, or make the current connection the default application connection by checking the **Set as default** checkbox.

A saved connection can be deleted by using the **X** button at the right of the connections list.

5.1.3.3 *Connecting to the Versa Router™ 1000 Quark*

In order to connect the NAMT to the Versa Router™ 1000 Quark, the PC hosting the NAMT must be connected in the same network segment (or subnet) as the Versa Router™ 1000 Quark.

Please see the Quick Start Guide for instructions on how to match the PC and Versa Router™ 1000 Quark IPv4s.

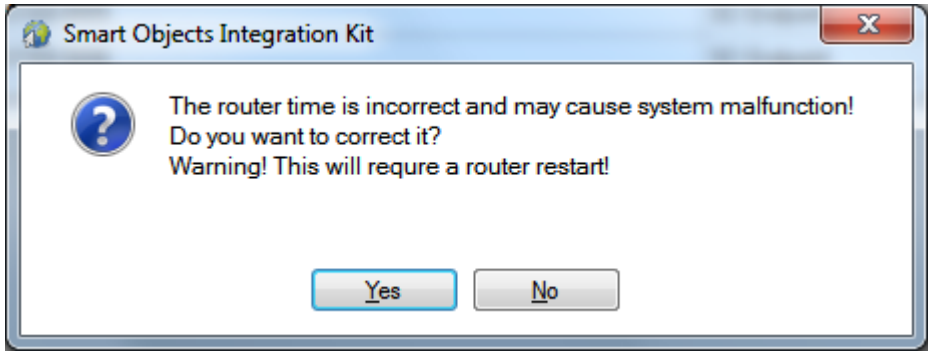
NOTE After powering on the Versa Router™ 1000 Quark, please wait 5 minutes before attempting to connect the NAMT to it.

When connecting the NAMT with the Quark, system behavior will differ depending on the PC and Versa Router™ 1000 Quark time. Please ensure the NAMT PC time is correct and does not suddenly change; otherwise, the system will behave inconsistently, leading to difficult-to-troubleshoot issues.

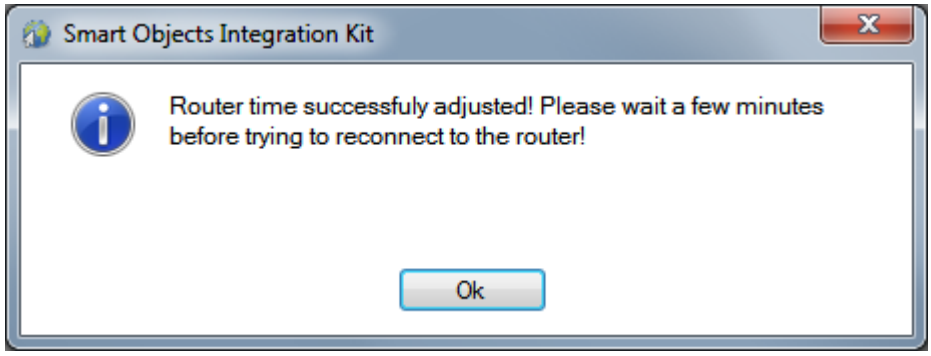
The Versa Router™ 1000 Quark time can be either:

- Not set (no Internet connectivity or incorrectly configured Internet connectivity)
- Properly set (by using Internet connectivity and automatically getting time from a time server, or previously set by a connecting NAMT)
- Set, but incorrect (usually using a connecting NAMT running on a PC with incorrect time)

If the Quark does not have the time set (no internet connectivity and no NAMT connected since last power on of the Quark), a message box is displayed asking the user's permission to correct the time on Quark. **Answering "Yes" is mandatory at this step; otherwise, the system will expose undefined behavior. Most notably, the Smart Objects will not join a system without proper time.**


Smart Objects Integration Kit
The router time is incorrect and may cause system malfunction!
Do you want to correct it?
Warning! This will requre a router restart!
Yes
No

After the user answers **Yes**, a message box will confirm the successful operation. The Quark will be restarted automatically and the network will re-form.

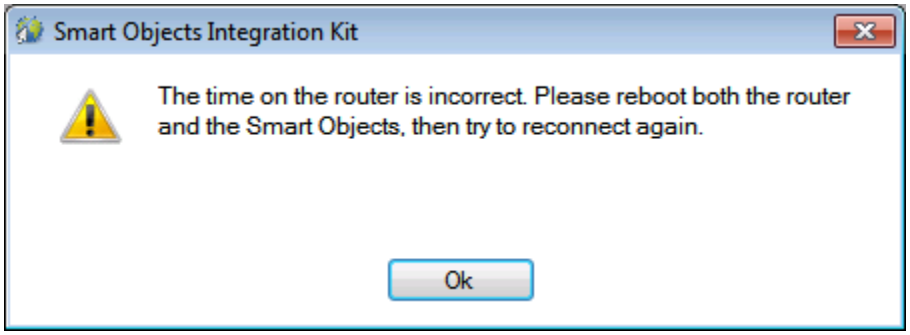

Smart Objects Integration Kit
Router time successfuly adjusted! Please wait a few minutes
before trying to reconnect to the router!
Ok

Please wait for about 5 minutes before re-connecting to the Quark after this operation.

NOTE

If the Quark time was previously set correctly, the "Router time incorrect" warning message box does not appear at all.

If the time on Versa Router™ 1000 Quark was set (via either the Internet or a previous NAMT connection) but then the PC time changed by more than one hour, the resulting system will have inconsistencies between the Smart Objects and the Quark, with undefined behavior. To prevent this, if the NAMT detects that the Quark has the time set but that the Quark time is too different (more than one hour apart) from the NAMT PC time, the NAMT will refuse to connect to the Quark and will recommend restarting the Quark and Smart Objects.

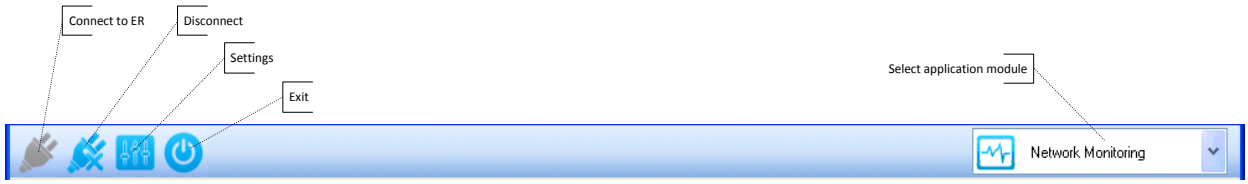

Smart Objects Integration Kit
The time on the router is incorrect. Please reboot both the router and the Smart Objects, then try to reconnect again.
Ok

Such a scenario can only occur if the user changes the time on the PC hosting NAMT, or if several PCs running NAMT are used to connect to the same Quark, with one of the PCs having incorrect time.

If you see the message above, please reboot the Versa Router™ 1000 Quark and all Smart Objects.

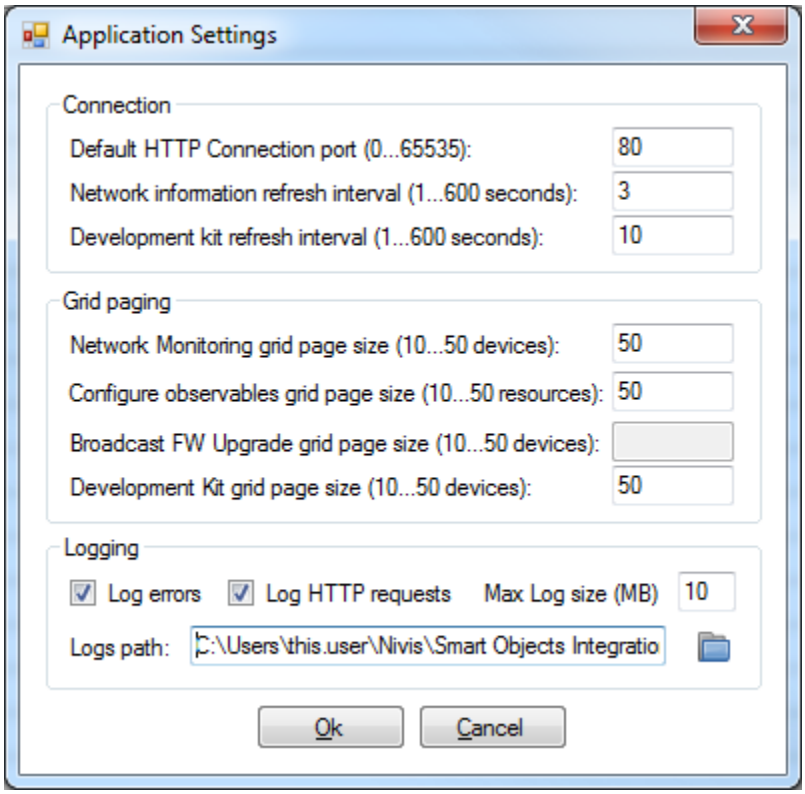
5.1.3.4 The main application window

After connecting or cancelling a connection to a Versa Router™ 1000 Quark, NAMT will display the main application window. From here, you can connect/disconnect a Versa Router™ 1000 Quark, manage the application settings, select the application module to use, or close the application by using the corresponding buttons in the toolbar.



5.1.3.5 Settings

Click on **Settings** to bring up the settings dialog.



The settings section allows you to:

- Change the http connection port (use the default 80).
- Set the refresh interval for network tab.
- Set the refresh interval for Development Kit tab.
- Change the number of displayed devices per page in Network Monitoring, Configure Observables, and Development Kit.
- Set logging options, to be used for troubleshooting purposes only when instructed by a Nivis representative.

We recommend always using the default refresh interval.

NOTE

The Broadcast FW Upgrade grid page size cannot be configured in this version.

Setting a refresh interval too small may affect the kit performance. Setting the refresh too high will result in slow data update in NAMT.

5.1.3.6 The Network Monitoring module

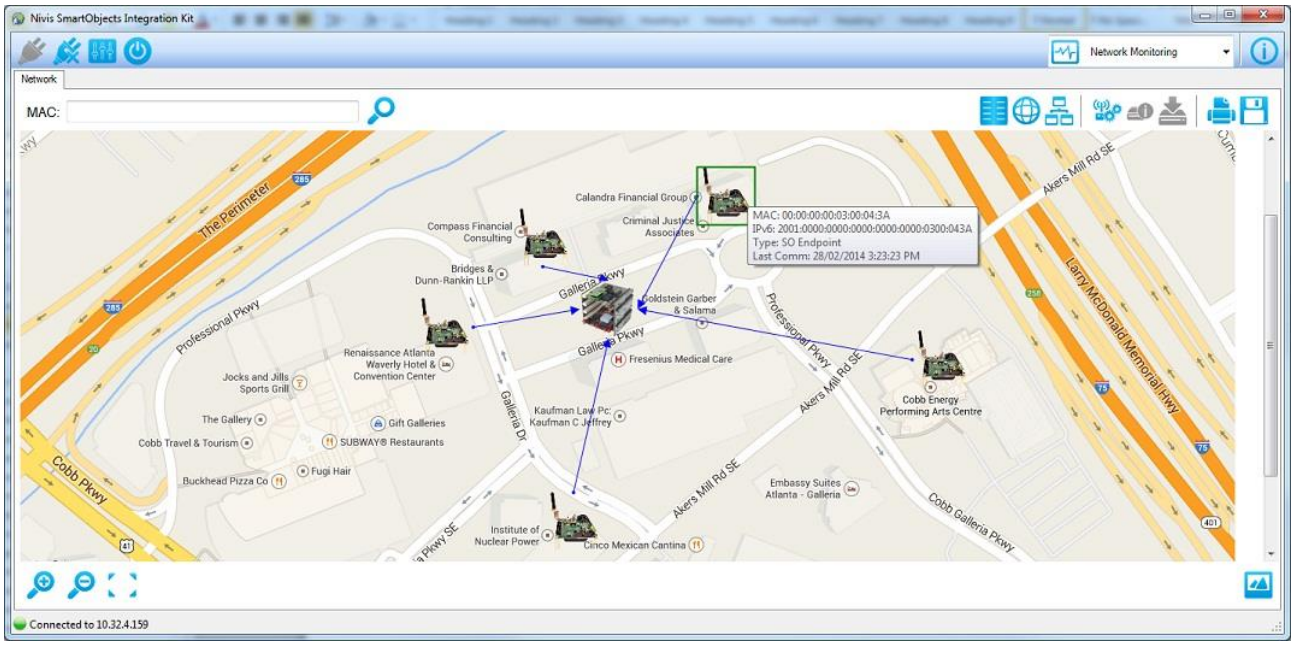
5.1.3.6.1 Overview

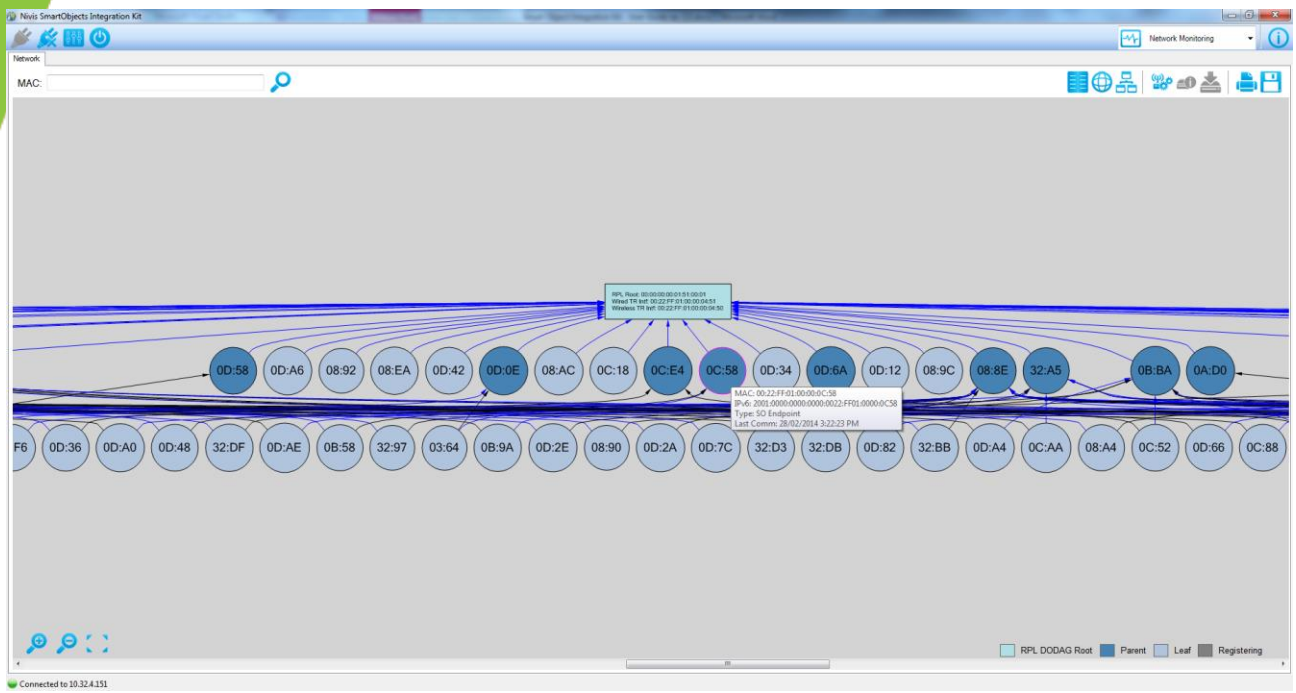
The **Network Monitoring** module provides the following network level monitoring and management functionality:

- Manage network settings – security level, communication parameters (channels map and hopping sequence)
- View the list of joined devices, details on the joined devices, channel statistics, and RPL Neighbor statistics for each device
- Update the firmware of the joined devices.

4

The user can see the list of joined devices in tabular form, search for a specific device, place the devices on a location map in map view, or see the logical topology of the network. The user can also print the current view or export the devices list to a file.

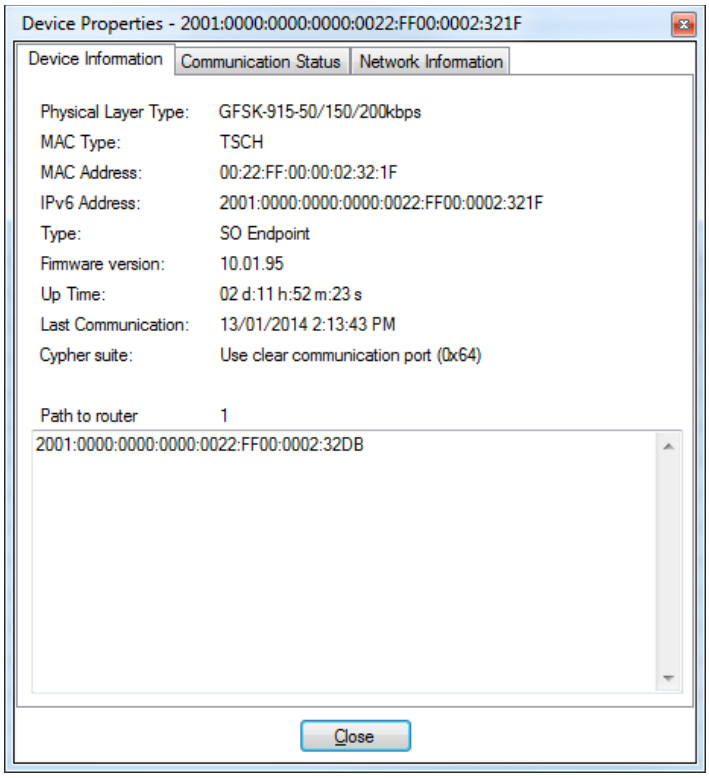




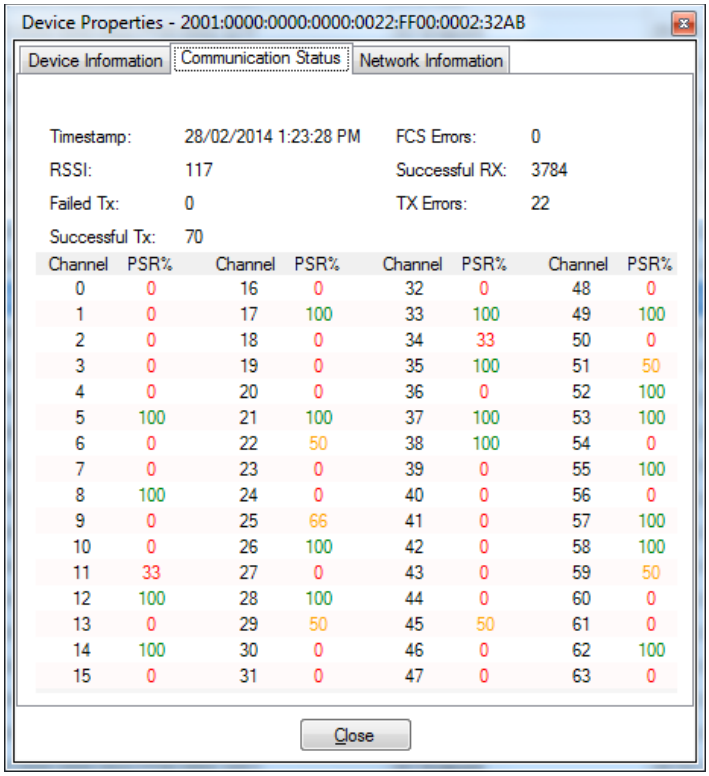
In the **Network Monitoring** module, the user can inspect the device properties by double-clicking on a device or by pressing the corresponding button in the module toolbar.

5.1.3.6.2 Device properties

The **Device Properties** window when the user clicks a device in the **Network monitoring** pane. The **Device Information**tab displays the details of the selected device.

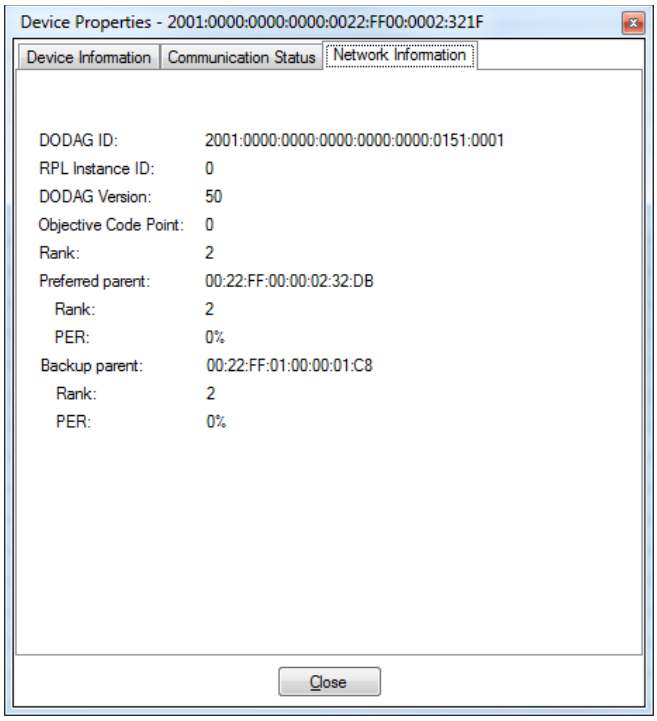

Device Properties - 2001:0000:0000:0000:0022:FF00:0002:321F
Device Information
Communication Status
Network Information
Physical Layer Type: GFSK-915-50/150/200kbps
MAC Type: TSCH
MAC Address: 00:22:FF:00:00:02:32:1F
IPv6 Address: 2001:0000:0000:0000:0022:FF00:0002:321F
Type: SO Endpoint
Firmware version: 10.01.95
Up Time: 02 d:11 h:52 m:23 s
Last Communication: 13/01/2014 2:13:43 PM
Cypher suite: Use clear communication port (0x64)
Path to router 1
2001:0000:0000:0000:0022:FF00:0002:32DB
Close

In the second tab, the device communications status is displayed as well, showing communication statistics and channels PSR. The PSR are color coded: green for good, orange for average, red for bad. Note that a value of 0 may mean “Unavailable: no statistics gathered.”



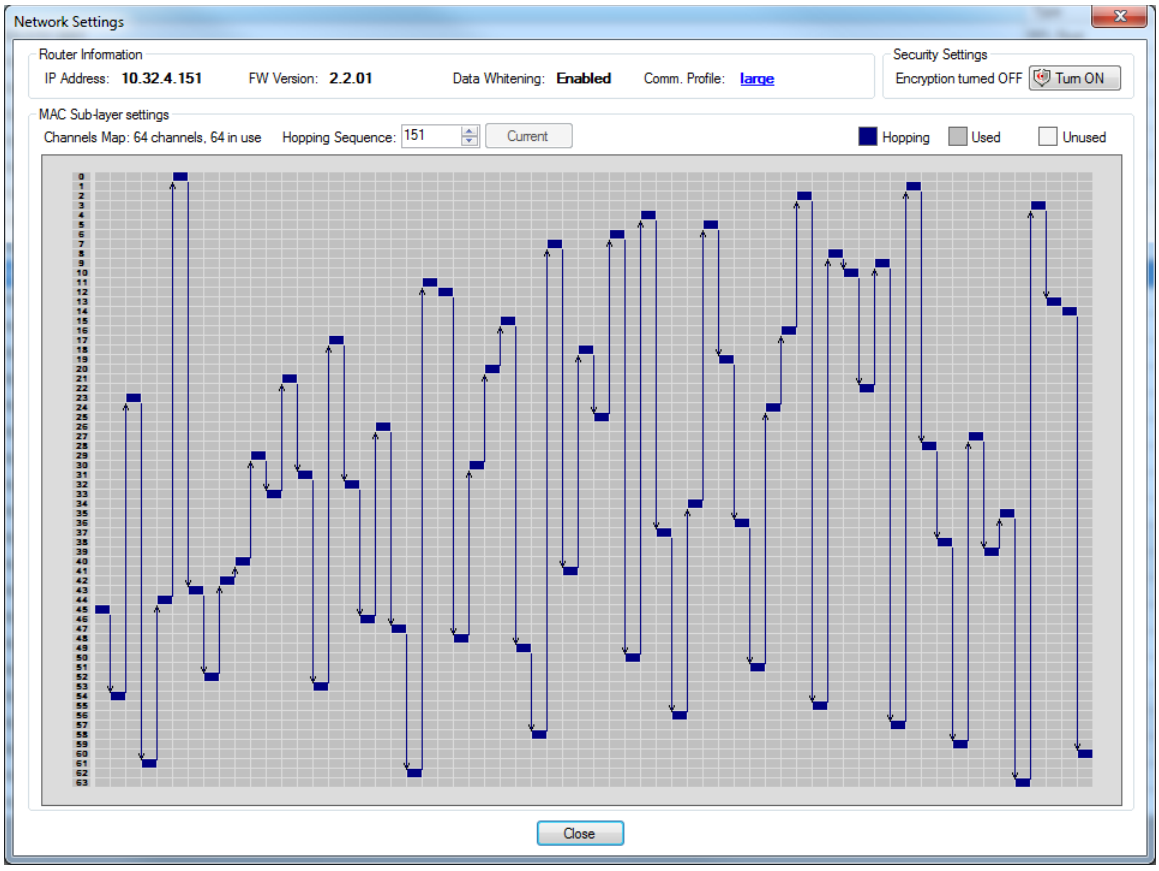
Channel	PSR%	Channel	PSR%	Channel	PSR%	Channel	PSR%
0	0	16	0	32	0	48	0
1	0	17	100	33	100	49	100
2	0	18	0	34	33	50	0
3	0	19	0	35	100	51	50
4	0	20	0	36	0	52	100
5	100	21	100	37	100	53	100
6	0	22	50	38	100	54	0
7	0	23	0	39	0	55	100
8	100	24	0	40	0	56	0
9	0	25	66	41	0	57	100
10	0	26	100	42	0	58	100
11	33	27	0	43	0	59	50
12	100	28	100	44	0	60	0
13	0	29	50	45	50	61	0
14	100	30	0	46	0	62	100
15	0	31	0	47	0	63	0

On the third tab of this window, the network information is displayed.



5.1.3.6.3 Network settings

The **Network settings** windows allow the user to inspect the router information and manage the network security level and channel hopping sequence.



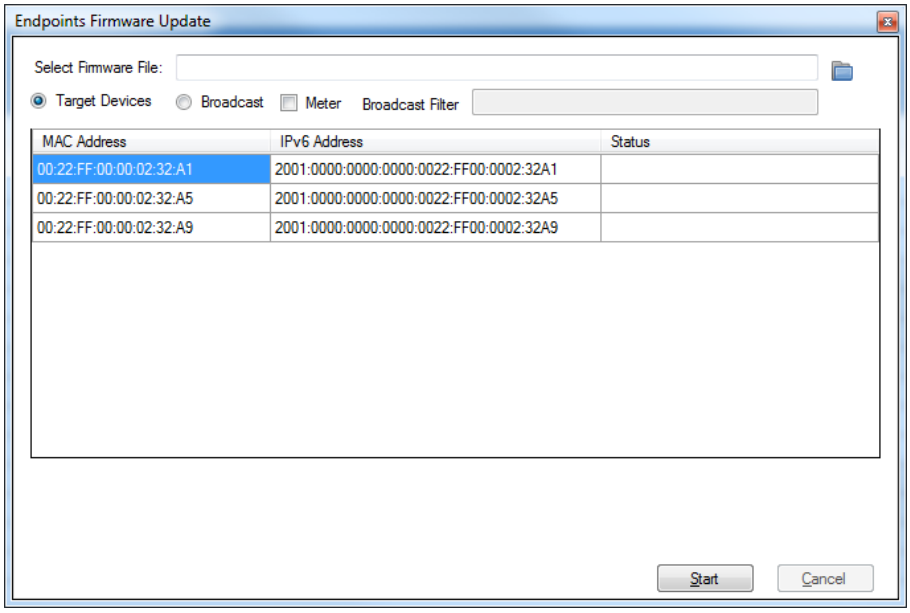
In order to view the current communication profile settings, the user must click the **Comm profile** link. A window containing the configuration file will be displayed.

In order to modify the network security level, click the **On/Off** button in the **Security Settings** group. The application will warn that this operation needs a network restart and will request your confirmation on continuing the operation.

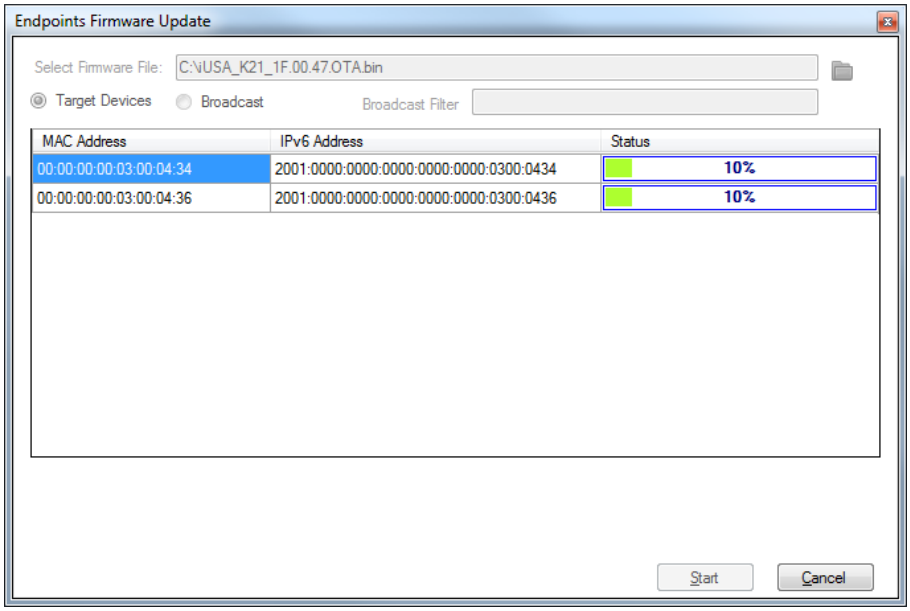
The hopping sequence used for network communication can be modified by using the up/down buttons in the hopping sequence edit box, then clicking **Apply**. The application will request confirmation to perform the change, as this requires a router restart.

5.1.3.6.4 Firmware update

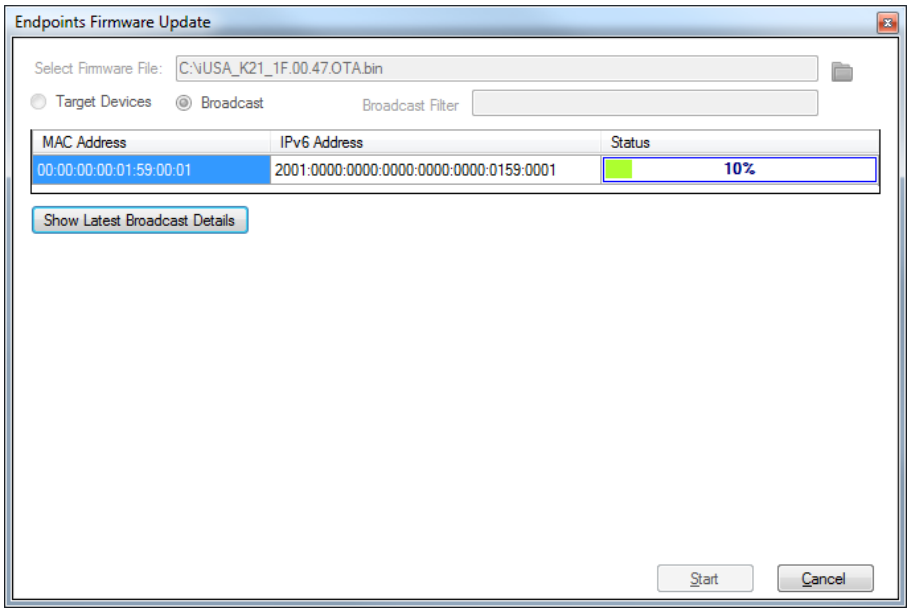
The **Firmware update** function allows the OTA update of the device firmware for the joined devices. In order to start a firmware operation on one or more devices, select one or more devices in the grid view (the view allows multi-select; use **Ctrl-click** for multiple selections), then click the **Firmware Update** button in the toolbar.



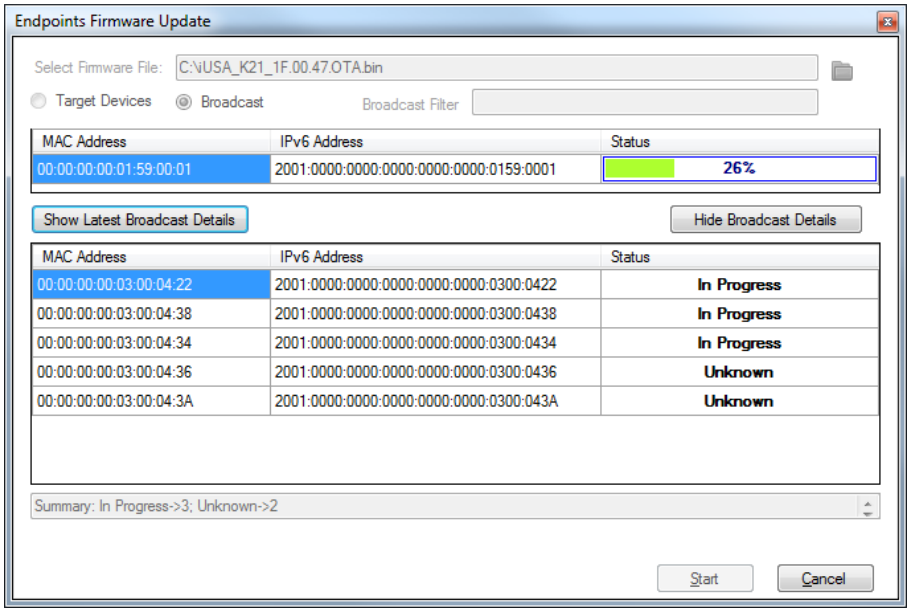
In the **Firmware Update** window, select the firmware file to be used, and then click **Start**. The status column will indicate the overall upgrade progress and the final status of the OTA upgrade.



To upgrade the whole network (or just a part of it), the **Broadcast** and **Broadcast filter** options should be used. The list with devices will contain only one element (the router) and will display the broadcast progress.



To view the progress of individual devices, the user must click **Show Latest Broadcast Details**. A list will be displayed with all network devices and their status. If a device has not responded yet to the broadcast command, its status will be *Unknown*. Bellow the device list, the broadcast summary will be displayed.



5.1.3.7 The Development Kit module

5.1.3.7.1 Overview

The Development Kit module has two views: the Dashboard view and the Grid view. The Dashboard View monitors and displays three fixed parameters of the Smart Object. The Grid View shows all COAP-modeled resources of a Smart Object, including user-defined resources, allowing subscription to observable resources, and also shows the values reported by those resources (with history in case of subscribed COAP observables).

The Development Kit module initially shows the Dashboard view.

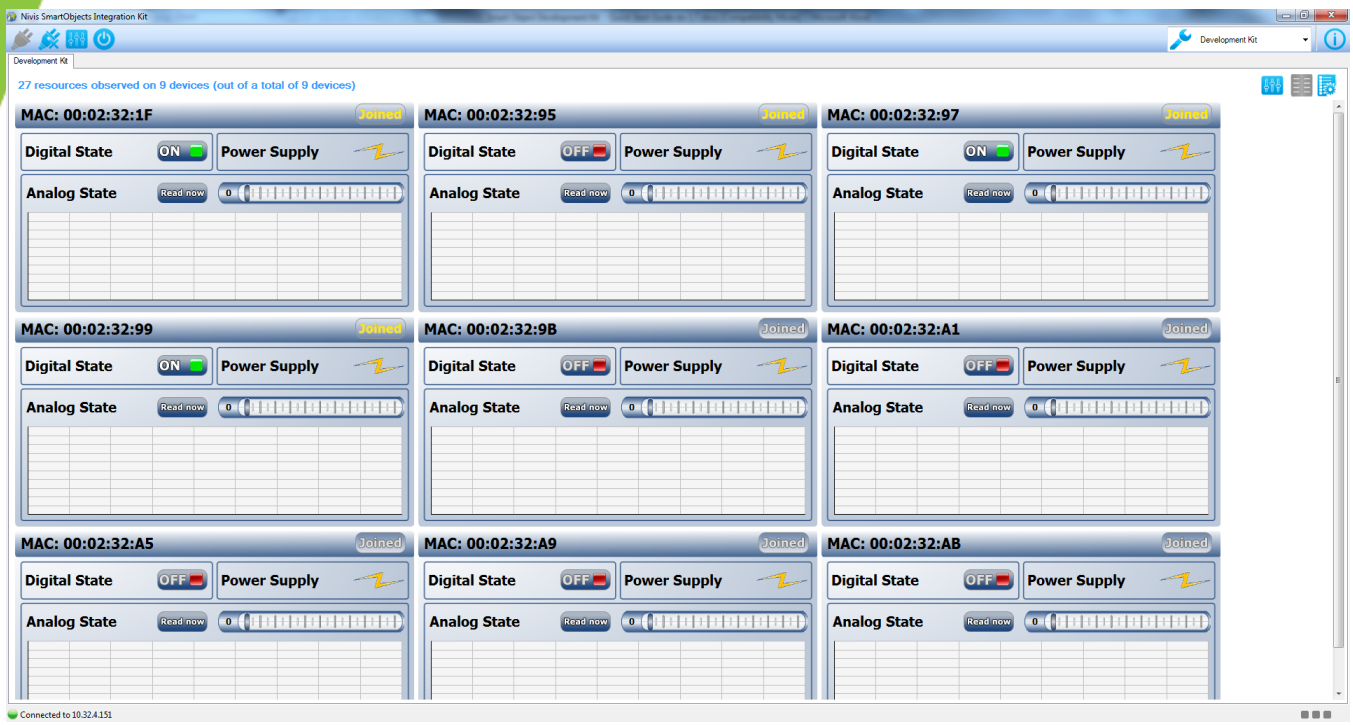
5.1.3.7.2 The Dashboard view

The SOEN monitors and periodically reports the parameters modeled as COAP resources listed in the table below.

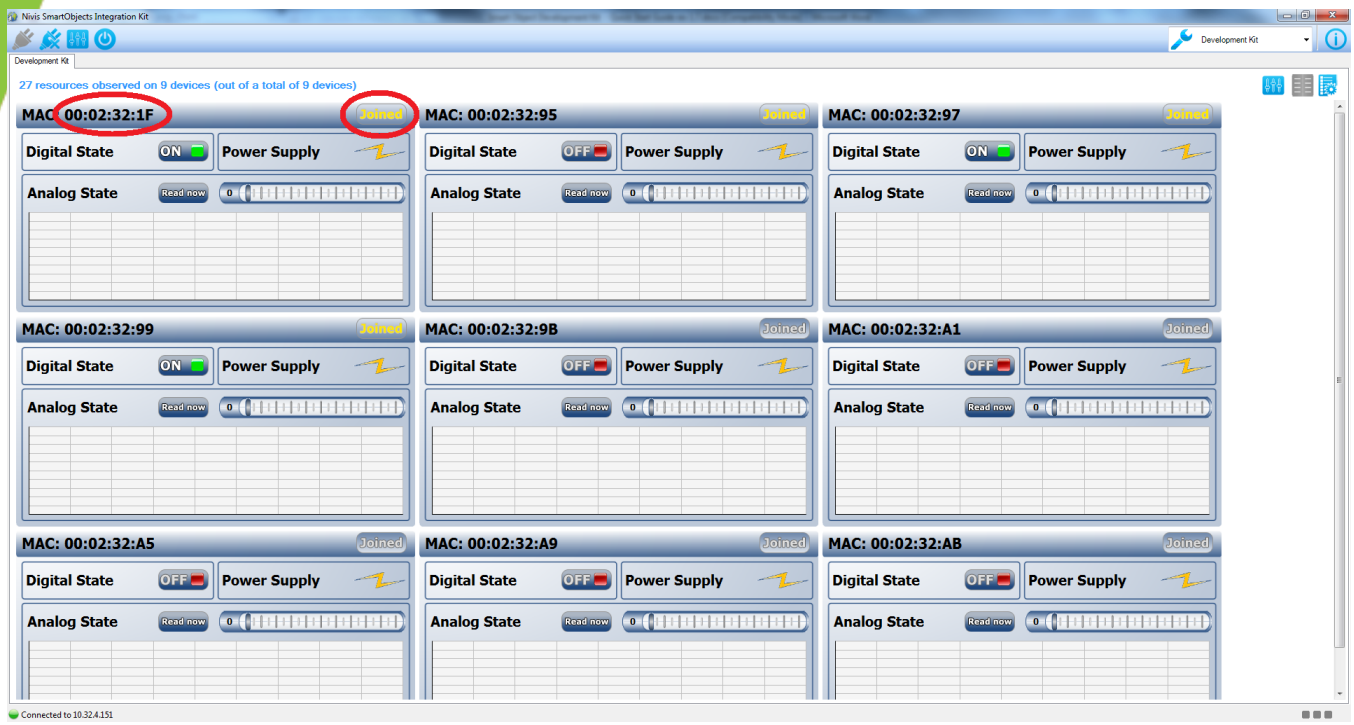
Parameter	URI	Power Pack Board Hardware Reference	Range	Reported
Digital ON/OFF	/app/sw	SW2	ON - OFF	Reported when status changes. Typical latency end-to-end (SOEN to NAMT display) is 3 seconds.
Analog stat	/app/ptm	R21	0 -100%	Reported when pot value varies by > 5%. Typical latency end-to-end (SOEN to NAMT display) is 3 seconds.
Battery	/app/batt	NA	Line power or battery power left (%)	Reported when battery value varies by > 5%. Typical latency end-to-end (SOEN to NAMT display) is 3 seconds.

Once the Smart Object network is up and running, a first evaluation scenario consists of simply observing changes in the three pre-defined application resources.

To enter the Development Kit screen of the NAMT, click the top-right drop-down in the main application window and select **Development Kit**.



Ensure in the NAMT that the SOEN is joined to the network (its MAC appears in the Development Kit, with status as Joined and yellow font). An example for a device with MAC ending with “00:02:32:1F” is shown below.



Using this screen, the user can monitor the current values for the three COAP-mapped application resources: potentiometer, status switch, and battery.

See

5.2 Set-up Quark / Cloud-Based NOC communication

Normal deployment scenario requires that the NMS - a subsystem of the Cloud-Based NOC – expose a public IPv4 address. The Quark may or may not have a public IPv4 address.

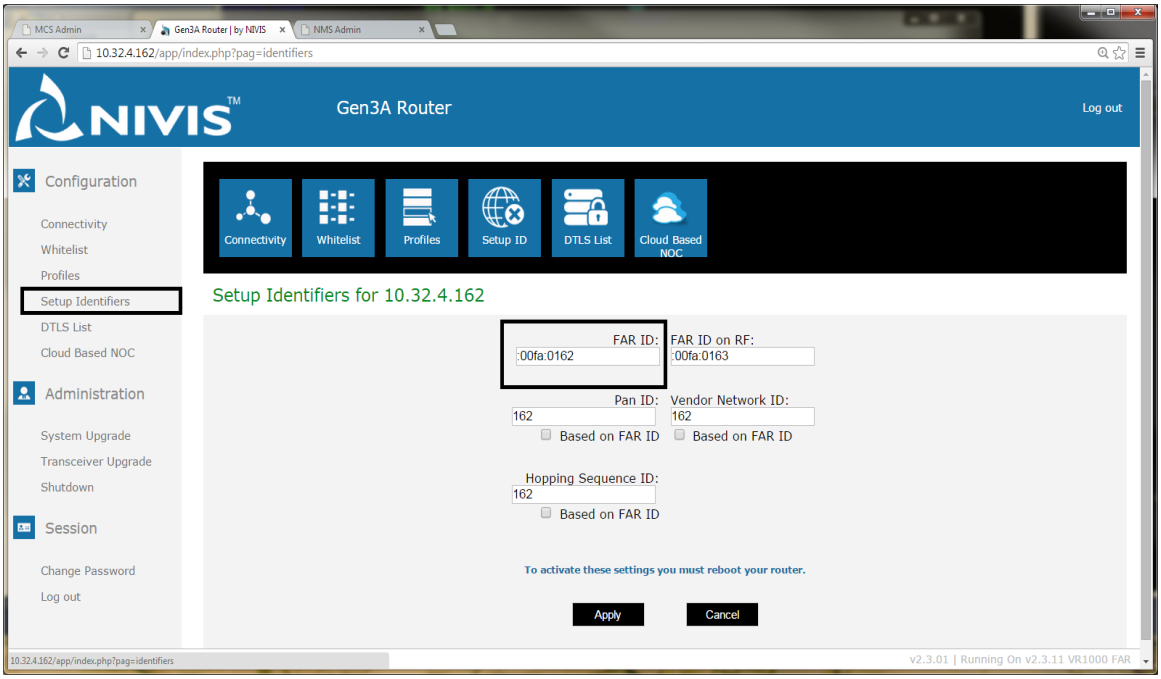
Recommended configuration is to use VPN for Quark-NMS connectivity.

Non-VPN communication is only acceptable for laboratory tests.

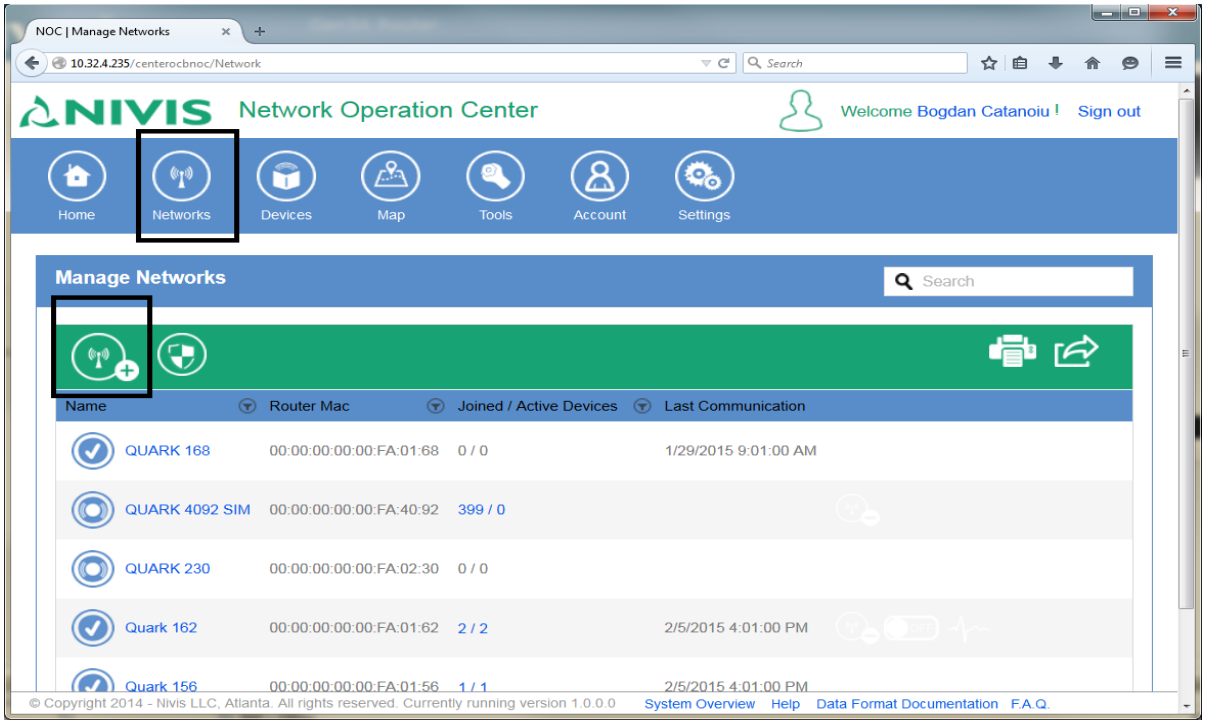
5.2.1 Add Quark in the Cloud-Based NOC Whitelist

Before trying to connect a Quark to the Cloud-Based NOC, the Quark FAR ID must be added to the Cloud-based NOC. The Cloud-Based NOC identifies networks using their controlling Quark.

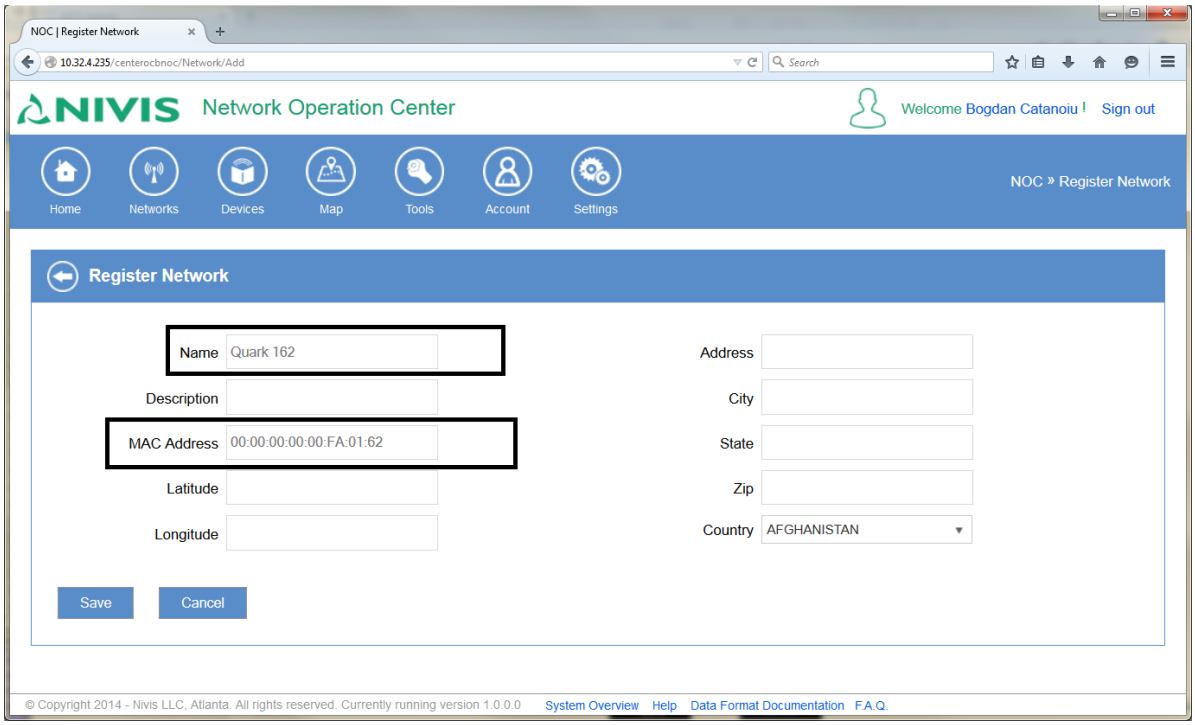
1. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
2. Go to “Setup Identifiers” page
3. Read or copy the FAR ID



4. Point a web browser to the Cloud-Based NOC website, enter credentials
5. Go to on “Networks” page
6. Press “Register Network” button


NOC | Manage Networks
10.32.4.235/centerocbnoc/Network
NIVIS Network Operation Center
Welcome Bogdan Catanoiu ! Sign out
Home
Networks
Devices
Map
Tools
Account
Settings
Manage Networks
Search
Name
Router Mac
Joined / Active Devices
Last Communication
QUARK 168 00:00:00:00:00:FA:01:68 0 / 0 1/29/2015 9:01:00 AM
QUARK 4092 SIM 00:00:00:00:00:FA:40:92 399 / 0
QUARK 230 00:00:00:00:00:FA:02:30 0 / 0
Quark 162 00:00:00:00:00:FA:01:62 2 / 2 2/5/2015 4:01:00 PM
Quark 156 00:00:00:00:00:FA:01:56 1 / 1 2/5/2015 4:01:00 PM
© Copyright 2014 - Nivis LLC, Atlanta. All rights reserved. Currently running version 1.0.0.0
System Overview Help Data Format Documentation F.A.Q.

7. Provide a network Name.
8. Use the Quark FAR ID identified at step 17 above to provide the “MAC Address”.
9. Press “Save”.



5.2.2 Set-up Quark and NMS communication using VPN [RECOMMENDED]

These steps below load Open VPN certificates into NMS and Quark. The steps are performed **only once** for NMS, and **only once** for each Quark, and then the Quark can be switched between “transparent” and “standalone” modes indefinitely without the need to re-load certificates.

Only if the root CA was changed on the Cloud-Based NOC (or when the Quark needs to connect to a different cloud-Based NOC, with a different root CA) the Quark will need to load a new certificate – generated using the new root CA.

NOTE

The certificate sets for Quark and NMS must be delivered in **.tar.gz** format.

The Quark certificate set is specific per Quark.

Do NOT attempt to use a certificate set built for a Quark on a different Quark.

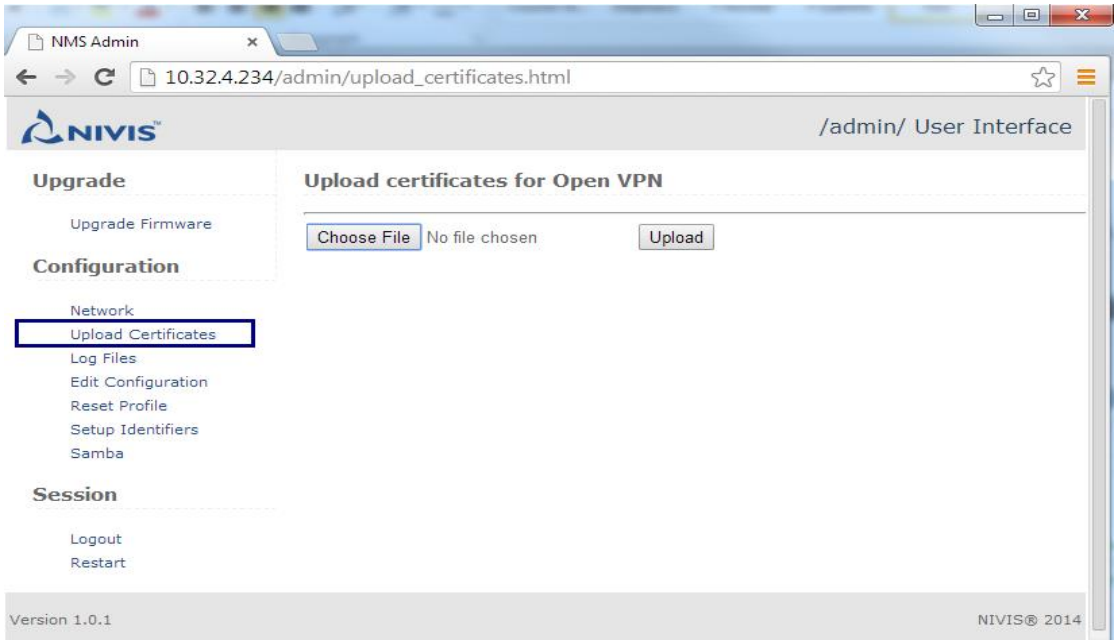
The GEN3 M1 Quarks are delivered with a set of certificates pre-installed. The certificate set is compatible only with Cloud-based NOC hosted by Nivis.

There’s no need to upload new certificate set on GEN3 M1 Quarks connecting to Nivis-hosted Cloud-based NOC.

In any other case, the certificate set needs to be uploaded to the Quark

5.2.2.1 *Set-up NMS VPN certificates*

10. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
11. Go to “Upload Certificates” page
12. Press “Choose File” button
13. Select the archive holding the NMS certificate set
14. Press “Upload” button

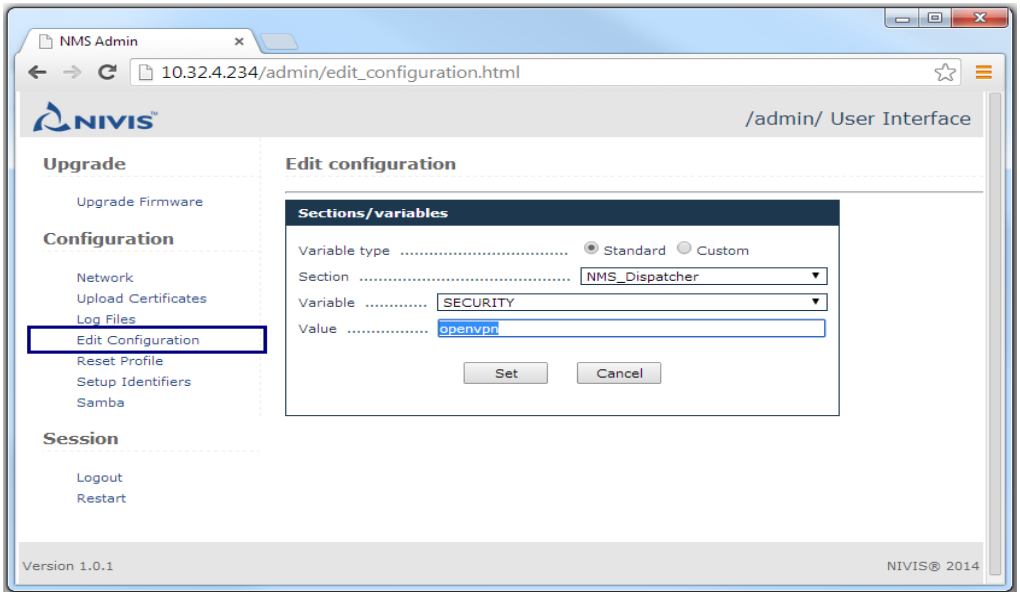


NOTE

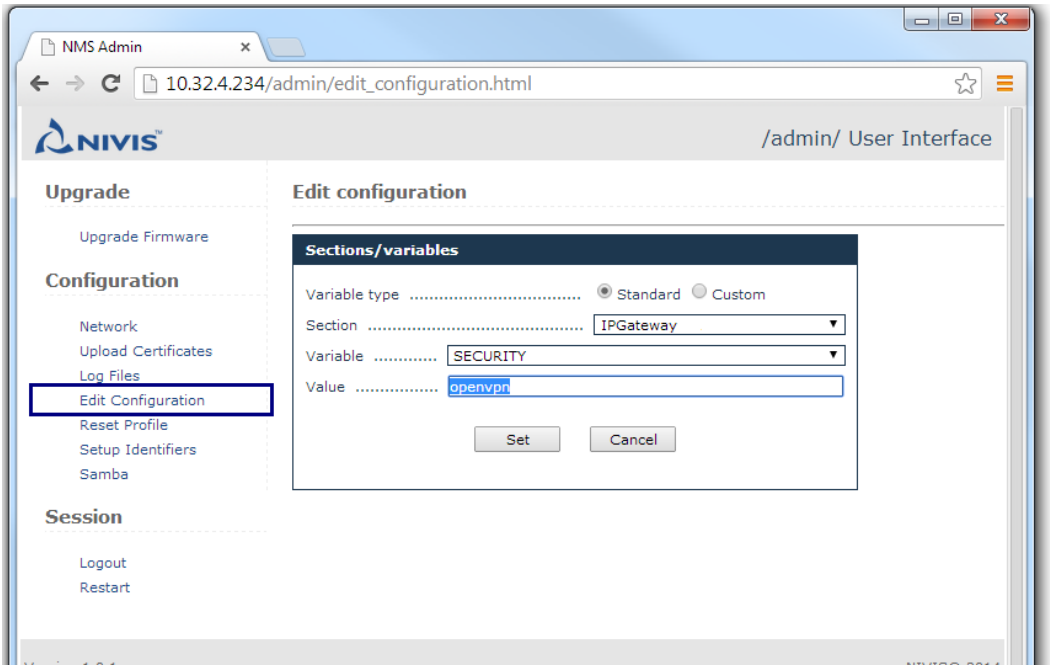
This steps has to be done only once, regardless of the number of subsequent switches of communication between “openvpn” and “none”

5.2.2.2 *Configure NMS modules to use OpenVPN*

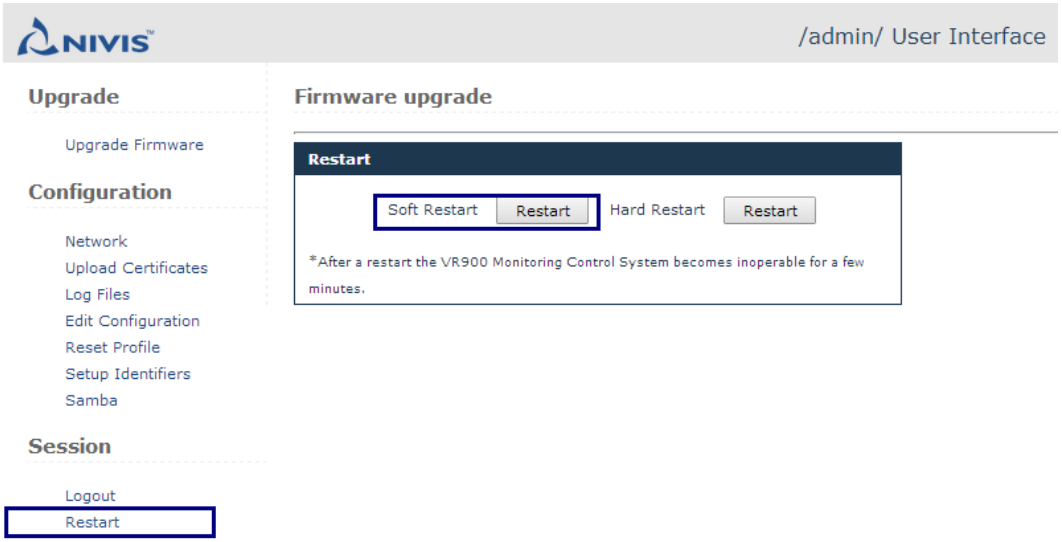
1. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
2. Go to “Edit configuration” page
3. Choose “NMS_Dispatcher” from “Section” drop-down
4. Choose “SECURITY” in “Variable“ drop-down
5. Make sure the “Value” edit reads “openvpn” (no quotes, not other char)



6. Choose “IPGateway” from “Section” drop-down
7. Choose “SECURITY” in “Variable“ drop-down
8. Make sure the “Value” edit reads “openvpn” (no quotes, not other char)

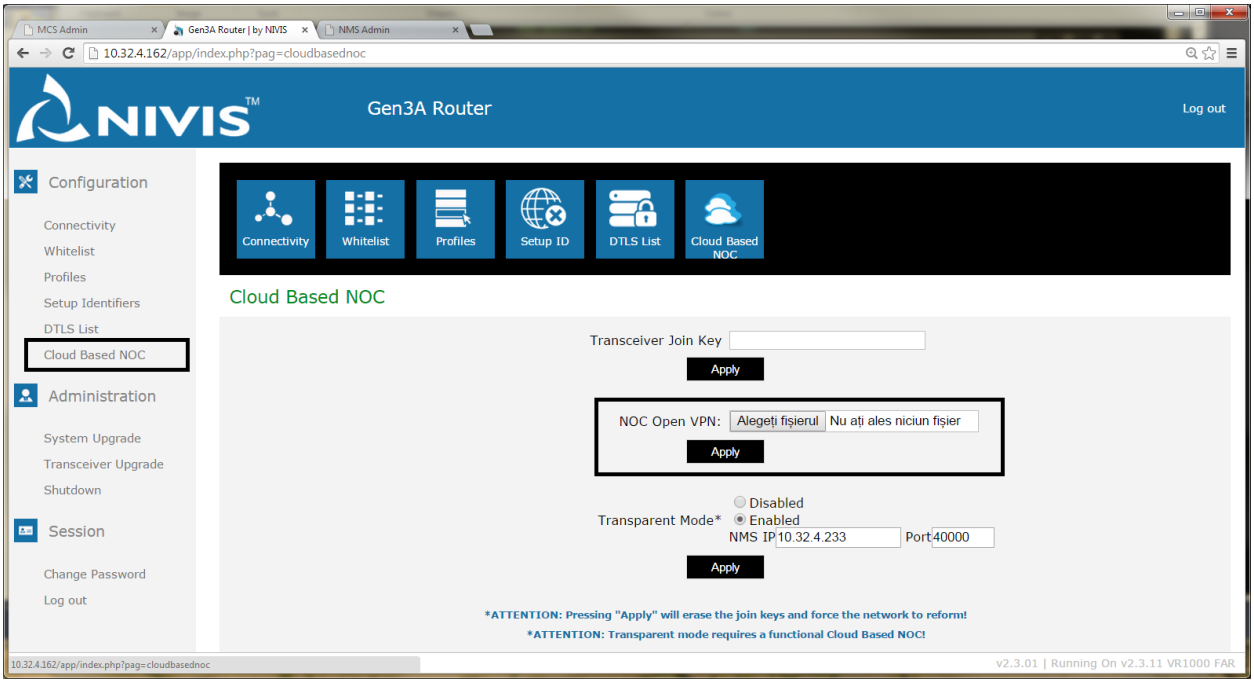


9. Go to “Restart” page
10. Perform a “Soft Restart”



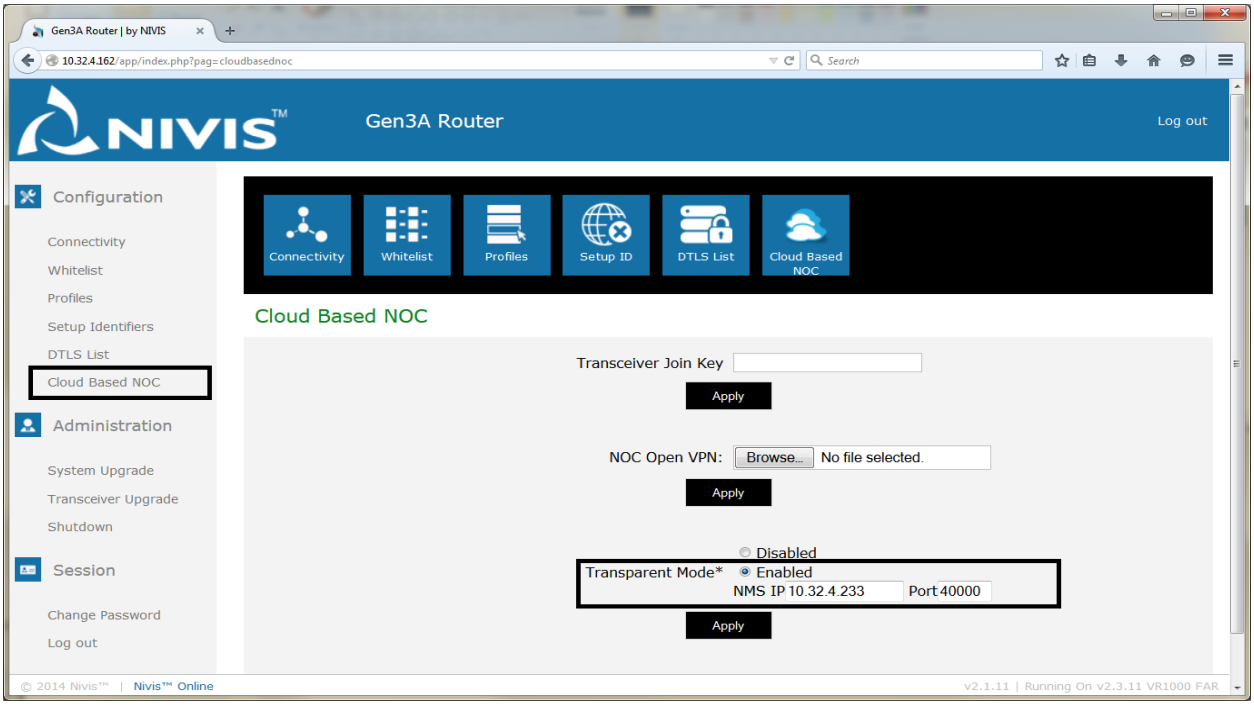
5.2.2.3 *Set-up Quark VPN certificates*

1. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
2. Go to “Cloud Based NOC” page
3. Press the “Choose File” button corresponding to ‘NOC Open VPN’ label
4. Browse for the archive with the Quark certificate set
5. Press the corresponding “Apply” button



5.2.2.4 *Set-up Quark in "transparent" mode (connect to NMS, use openvpn)*

1. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
2. Go to "Cloud Based NOC" page
3. Click to Enable the transparent mode
4. Fill in the NMS **Public IPv4**
5. Fill in the NMS port (default: 40000)
6. Press "Apply" button



NOTE

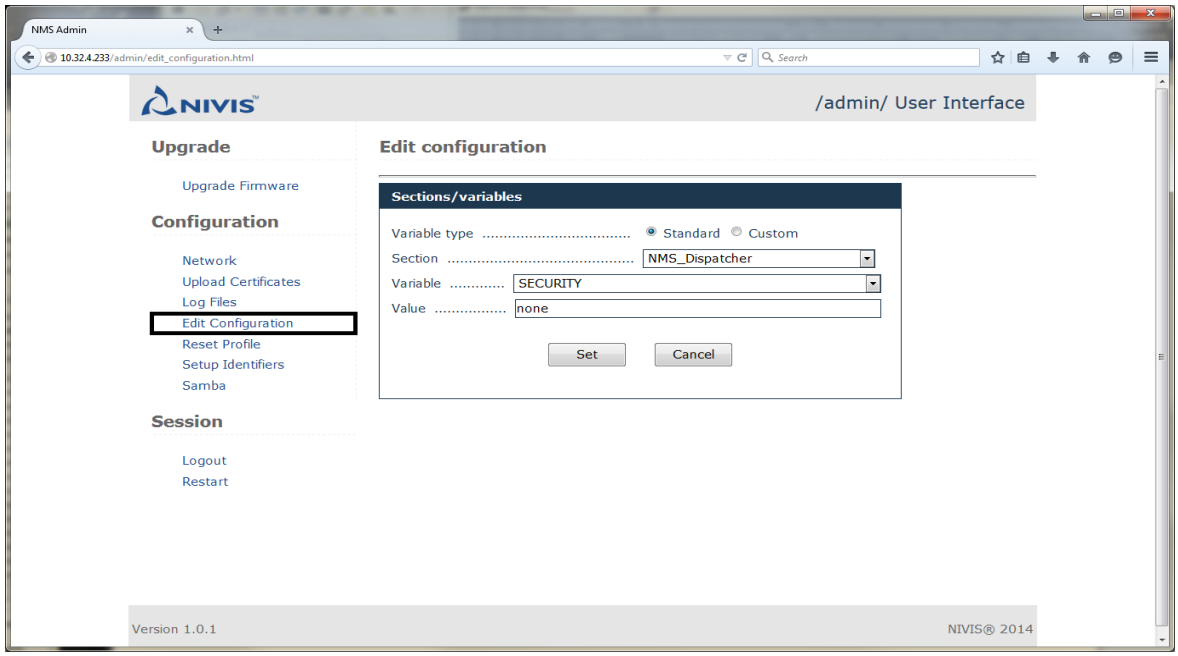
Enabling "transparent" mode on Quark automatically enables the "openvpn" communication for Quark relevant module (BHManager)

5.2.3 [Alternate] Set-up Quark-NMS plain-text communication (Use for lab tests only)

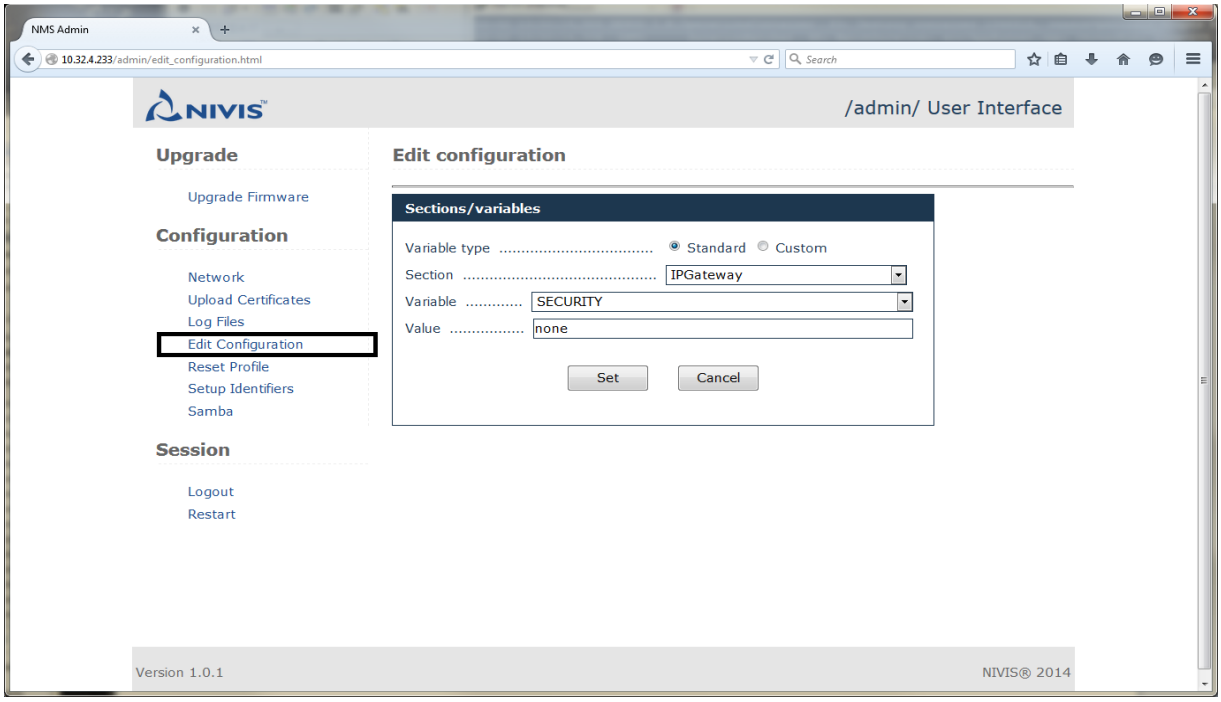
These steps below configure NMS and Quark for non-VPN (open/plaintext) communication. The steps are performed **only once** for NMS, but **each time** the Quark is switched between "standalone" and "transparent" mode, because setting the "transparent" mode automatically enable VPN communication and this step disables it.

5.2.3.1 Configure NMS modules to use open communication

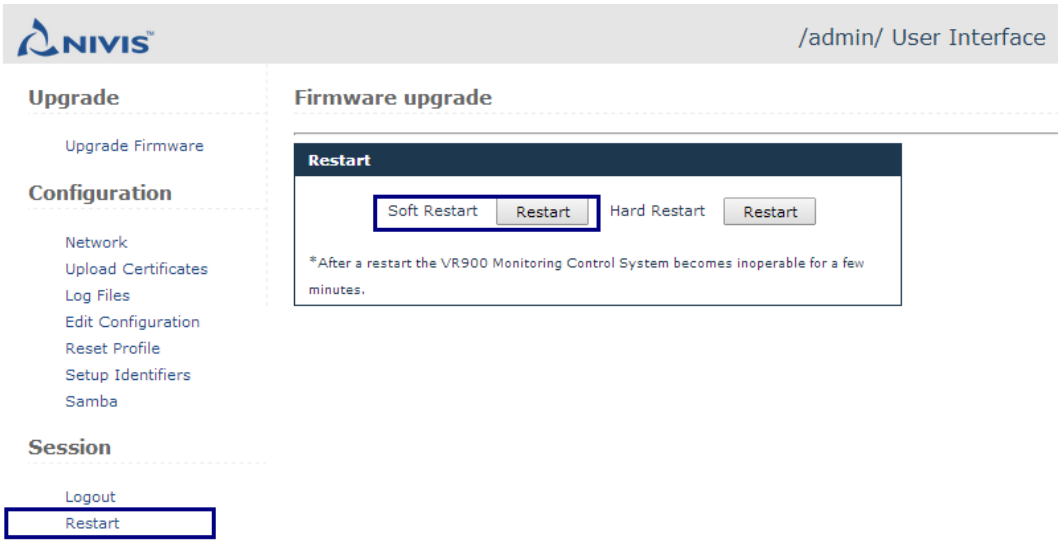
1. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
2. Go to "Edit configuration" page
3. Choose "NMS_Dispatcher" from "Section" drop-down
4. Choose "SECURITY" in "Variable" drop-down
5. Set the field "Value" to "none" (no quotes, not other char)



6. Choose “IPGateway” from “Section” drop-down
7. Choose “SECURITY” in “Variable“ drop-down
8. Set the field “Value” to “none” (no quotes, not other char)

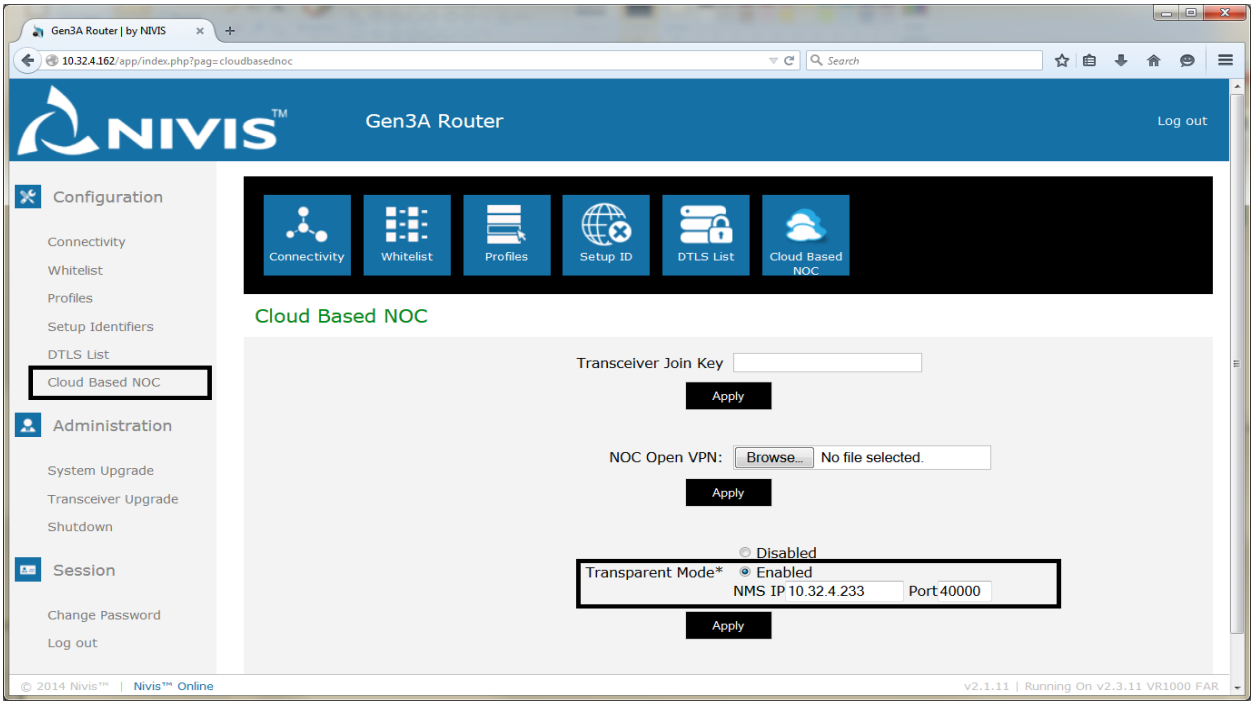


9. Go to “Restart” page
10. Perform a “Soft Restart”



5.2.3.2 *Set-up the Quark in "transparent" mode – connect to the Cloud-based NOC*

1. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
2. Go to "Cloud Based NOC" page
3. Click to Enable the transparent mode
4. Fill in the NMS **Public IPv4**
5. Fill in the NMS port (default: 40000)
6. Press "Apply" button



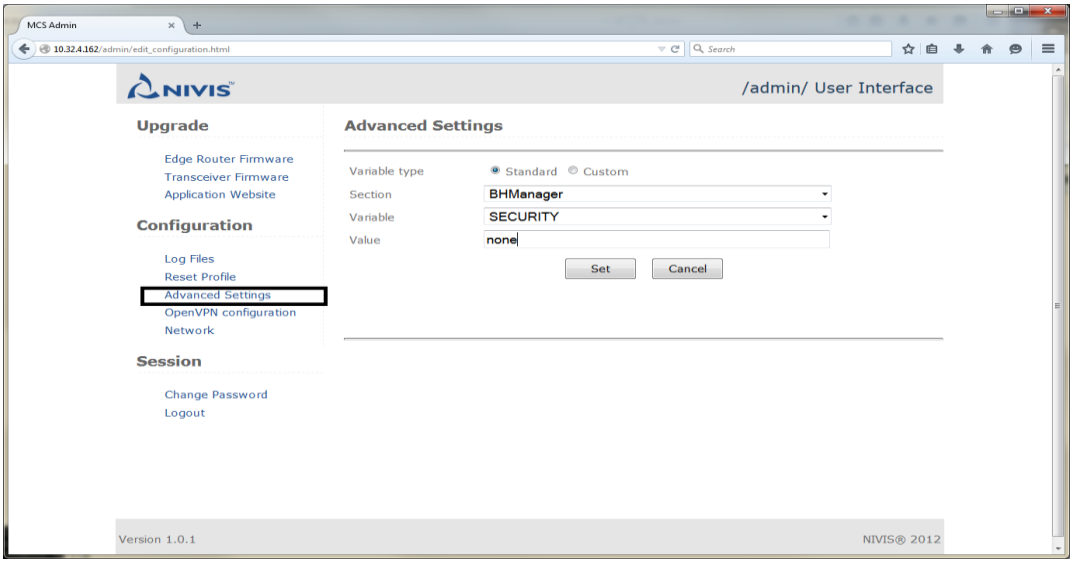
NOTE

Enabling "transparent" mode on Quark automatically enables the "openvpn" communication for Quark relevant module (BHManager). Sine this step uses open communication, the next step is necessary to set the communication back to "open" for BHManager

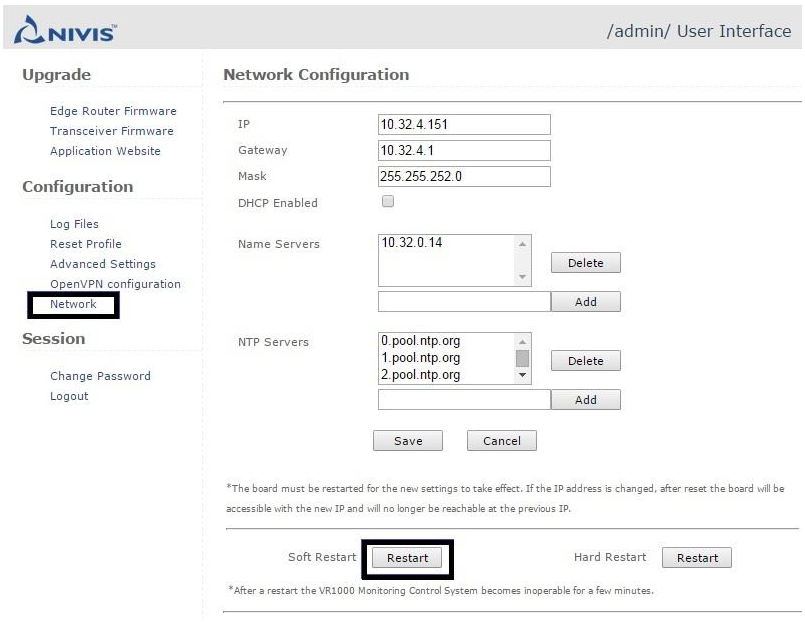
5.2.3.3 *Configure Quark modules to use open communication*

This step must be performed **after** setting the "transparent" mode, as setting the transparent mode automatically configures the Quark modules to use OpenVPN. After setting the "transparent" mode, the Quark software restarts rendering the web interface inaccessible for few tens of seconds. Wait until the web interface is accessible.

1. Point a web browser to the Quark /admin/ website (http://<quark_ip>/admin/), enter credentials
2. Go to "Advanced settings" page
3. Choose "BHManager" from "Section" drop-down
4. Choose "SECURITY" in "Variable" drop-down
5. Set the field "Value" to "none" (no quotes, not other char)



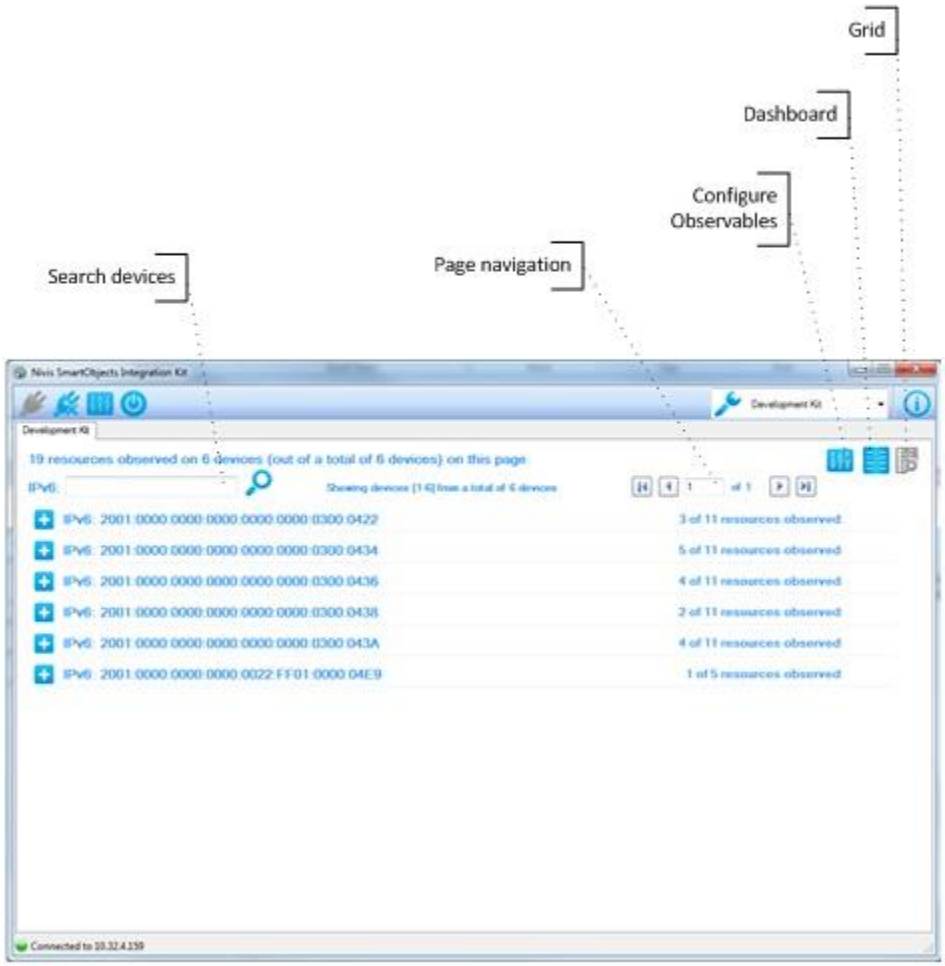
6. Go to "Network" page
7. Perform a "Soft Restart" to activate the changes



Use Cases for additional details.

5.2.3.3.1 The Grid view

In the Development Kit module, the user can also configure COAP observables for each device. Click on the **Grid View** icon in the top right of the window to switch to the Grid View.



The Grid View shows for each device the total number of COAP resources and the number of currently observed COAP resources. It also shows a summary with total number of COAP resources exposed by all devices. Initially, all of the devices are shown collapsed.

Click on the expand (“+”) sign next to a device to expand the view of that device.

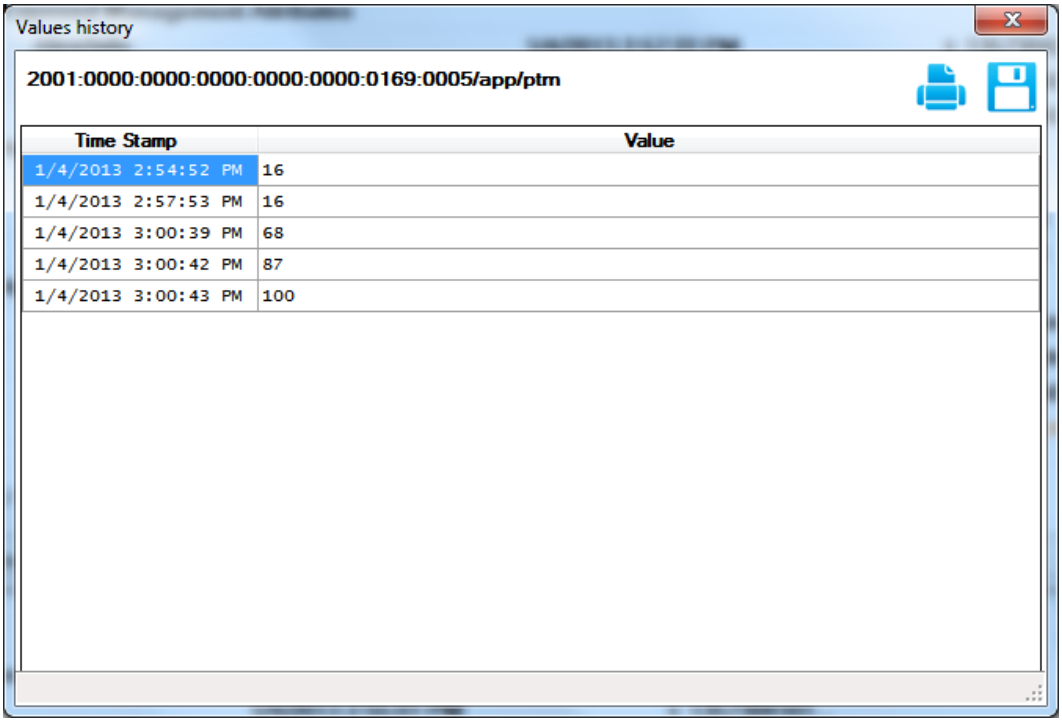
The following controls are available:

- Display a summary of number of devices and resources, total per system
- Display the total number of resources and observed resources for each device
- Expand a device for additional controls:
    - Show all resources
    - Show last COAP communication time
    - Highlight last received value (if belongs to device currently expanded)
    - Access values history
    - Show whether the resource is observable or not
    - Show whether the resource is observed or not
    - Show whether the resource is in an intermediary state or not

If there is an Application Processor connected to the Smart Object and the Application Processor has defined user-defined COAP resources, they will show in NAMT as any other COAP resource. See the “Nivis Smart Object API Integration Manual rev 2.4” for instructions on how to integrate user-defined COAP resources.

5.2.3.3.2 The values history

Clicking on the **History** icon brings up the values history for a COAP resource. If the resource is not observable or not observed, the history has a single reading. If the resource is (or was) observed, the history may show more than a single value, if more than one was reported by the SO.



NOTES

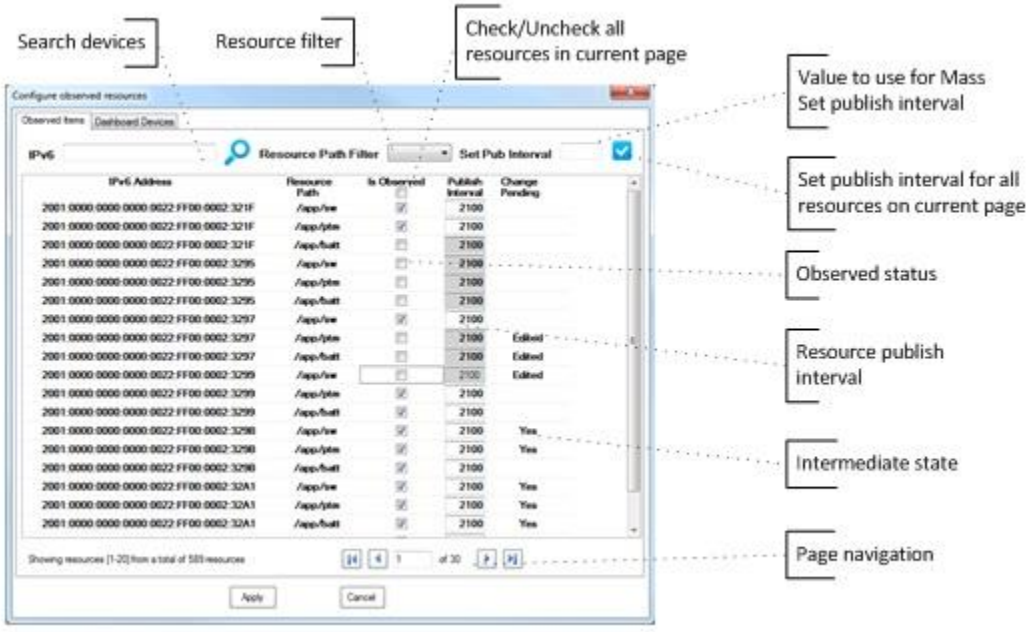
The values history is stored on the Quark and retrieved by NAMT when the **History** button is clicked.

Initially, the History page shows only the last value read and initiates a history load. While loading the history, the text "**Loading…**" appears in the lower left corner of the page.

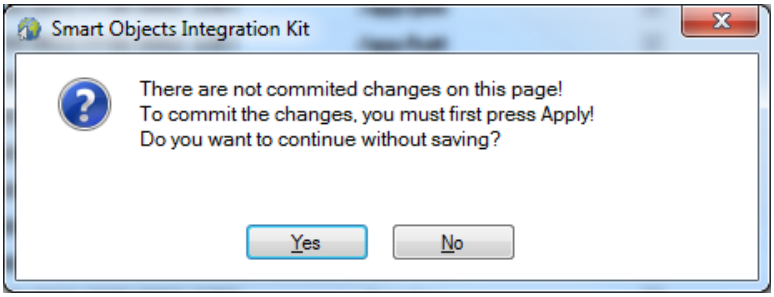
It may take a few seconds (depending on the network connection speed) until the history is fully loaded; then the lower left corner of the page will show "**Loading Finished**" and the page will show multiple history values and their timestamps.

5.2.3.3.3 Subscription to COAP observables

To subscribe or unsubscribe to COAP observables resources, open the **Configure Observables** window. Check/uncheck the checkbox on the right side of each resource, and then click **Apply** to apply the changes (the status for multiple observable resources can be changed at one time). From this window, the user can also set the publish interval of the observed resources.


Search devices
Resource filter
Check/Uncheck all resources in current page
Value to use for Mass Set publish interval
Set publish interval for all resources on current page
Observed status
Resource publish interval
Intermediate state
Page navigation

If the user goes to another page without saving the changes, the application asks for confirmation.

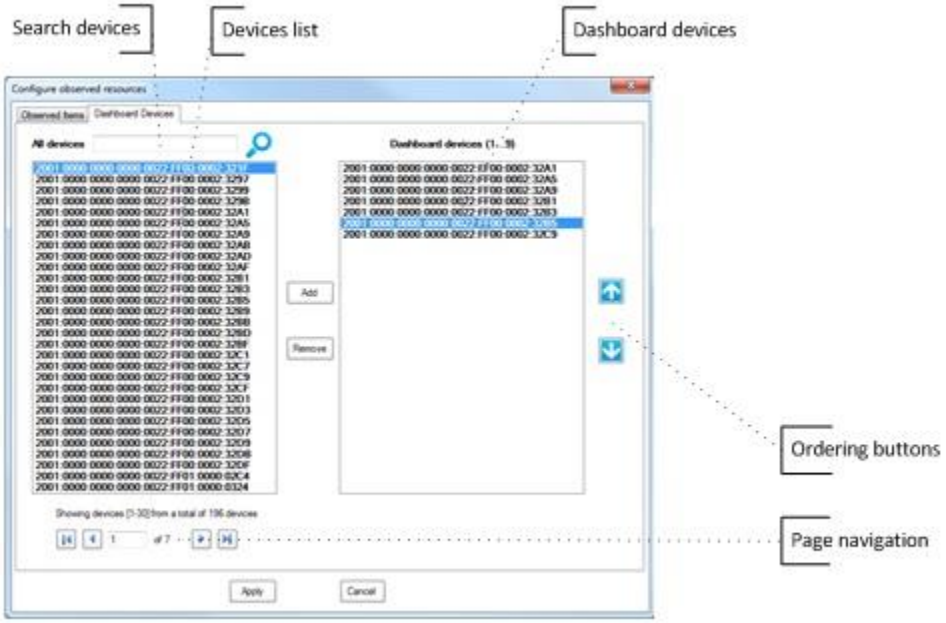

Smart Objects Integration Kit
There are not commited changes on this page!
To commit the changes, you must first press Apply!
Do you want to continue without saving?
Yes
No

After confirmation, the subscriptions/un-subscriptions to the COAP observable resources are submitted to the devices.

WARNING It may take several seconds to apply the settings. **Do NOT** edit the resource subscription while the subscription process is not finished (**Change Pending** column shows **Yes**).

5.2.3.3.4 Configure Dashboard devices

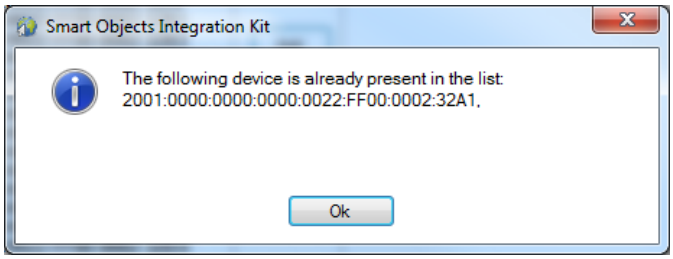
The user can choose the devices that will be displayed in the Dashboard. This feature can be accessed via **Development Kit → Configure Observables →Dashboard Devices**.



On the left is displayed the list with all devices (or with some particular devices if the user uses the search feature). This list supports multiple select. Also, because this list can contain hundreds of devices, the user can use the page navigation controls situated at the bottom of the window.

On the right is displayed the list with the devices configured to appear on the Dashboard. This list can be ordered by using the arrow-shaped buttons from the right.

To add a device on the Dashboard, the user must select a device from left-side list and then click **Add**. Keep in mind that the maximum number of devices that can be shown in the Dashboard is 9. Also, if the user wants to add a device that is already configured on the Dashboard, the application will display a warning window.



To remove a device from the Dashboard, the user must select a device from the right-side list and click **Remove**.

6 Upgrading the Development Kit

6.1 Overview

The **SO stack firmware** can be upgraded over the serial line, over-the-air from NAMT, and over-the-air from CBNOC.

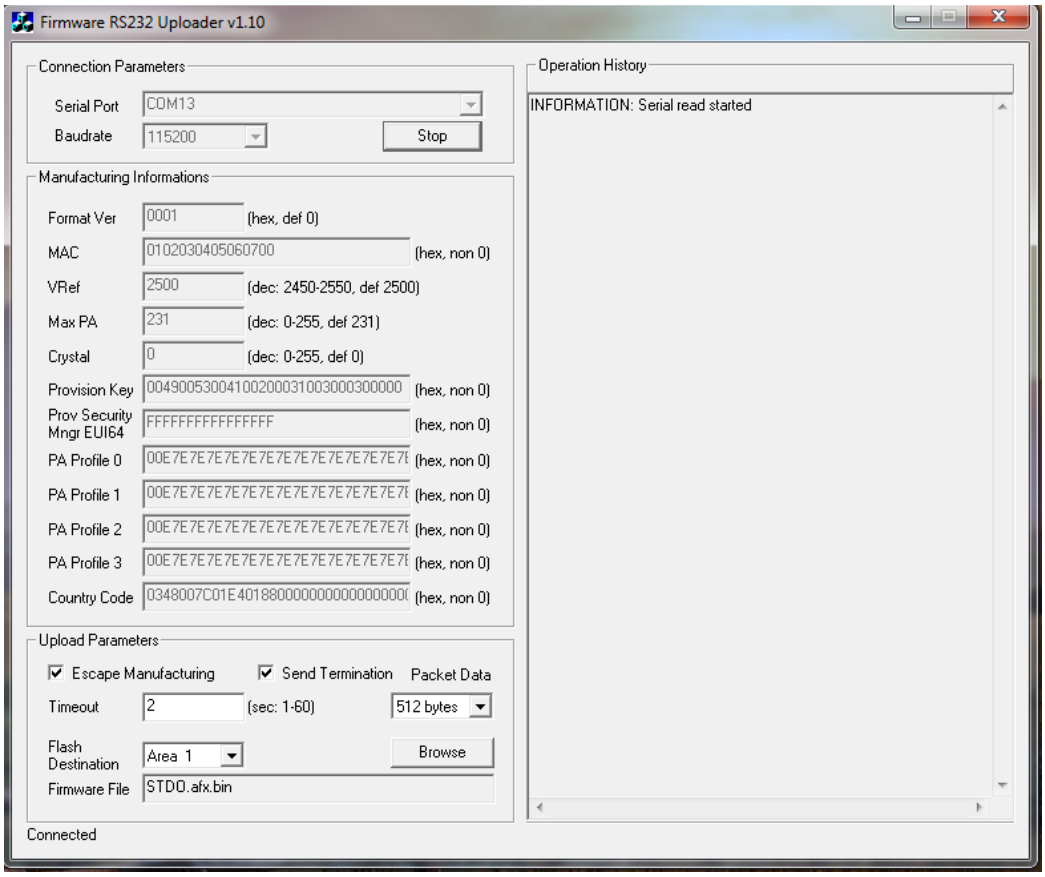
The **Quark software** can be upgraded by using the Quark website, the Quark /admin/ interface, and using the Cloud-based NOC website.

The firmware running on the Quark **Transceiver** can be upgraded by using the Quark website, the Quark /admin/ interface, and using the Cloud-based NOC website.

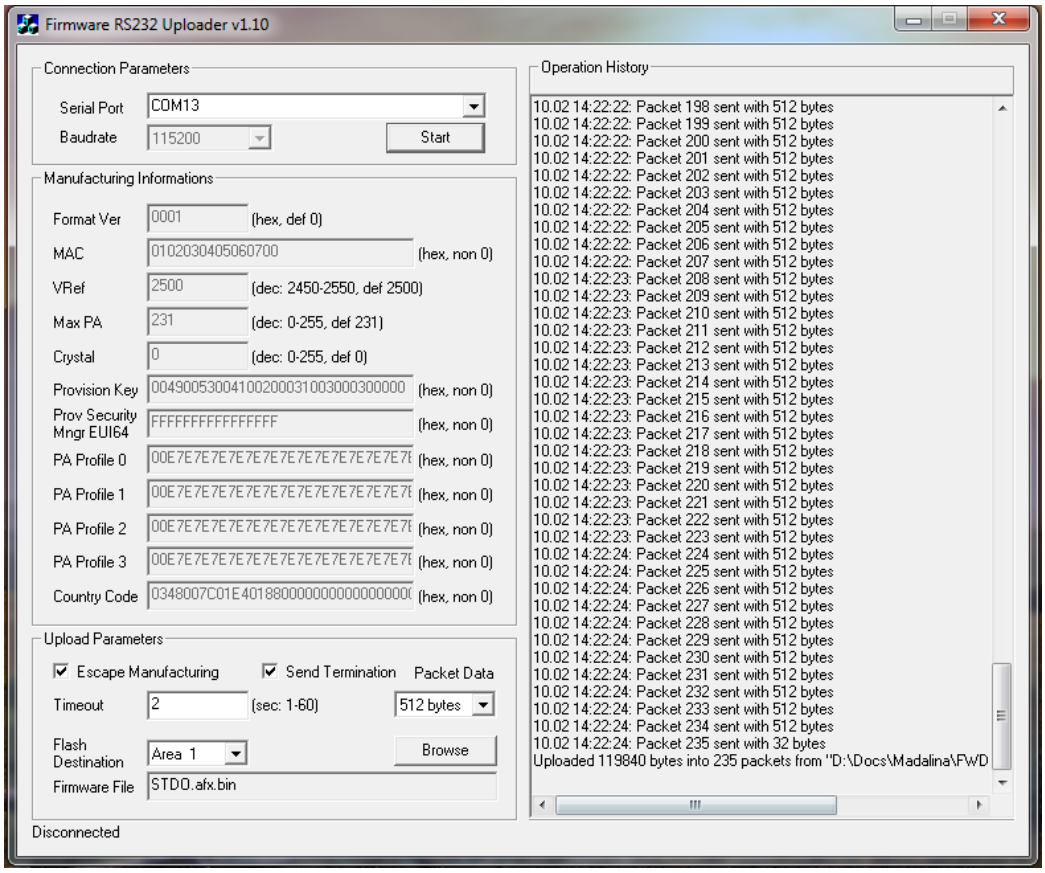
The upgrade via Cloud-based NOC website can only be performed if the Quark is running on transparent mode, connected to the Cloud-based NOC.

6.2 Upgrading the Smart Object stack firmware using serial line

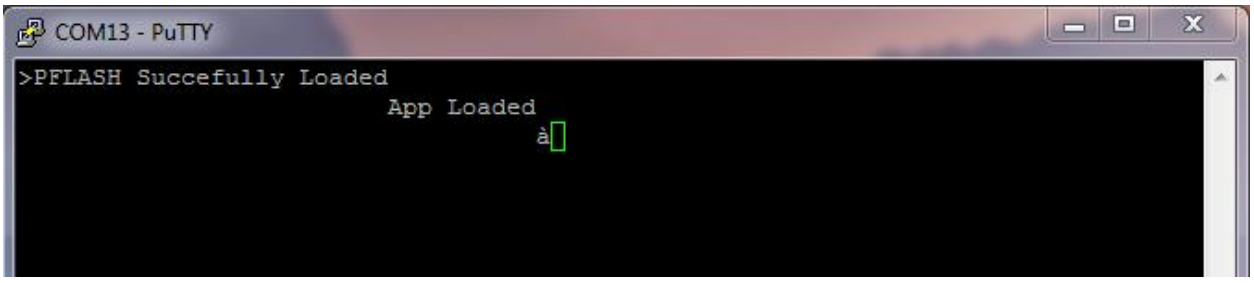
1. Upload a binary image with the Firmware RS232 Uploader (Upload2Serial MFC.exe).
    - Connection Parameters:
        - Set the Serial Port
        - Baud rate : 115200
    - Upload Parameters:
        - Set:
    - Timeout in seconds: any value between 1 and 60
    - Packet Data:512bytes
    - Send Termination: Enabled
    - Flash Destination: Area1, Area2, or Area 3



2. Click Browse... and select the binary (file .afx.bin).
3. Close the serial port if it is already in use.
4. Click Start.
5. Reset the device.


Firmware RS232 Uploader v1.10
Connection Parameters
Serial Port
COM13
Baudrate
115200
Start
Manufacturing Informations
Format Ver
0001
(hex, def 0)
MAC
0102030405060700
(hex, non 0)
VRef
2500
(dec: 2450-2550, def 2500)
Max PA
231
(dec: 0-255, def 231)
Crystal
0
(dec: 0-255, def 0)
Provision Key
(hex, non 0)
Prov Security Mngr EUI64
FFFFFFFFFFFFFFFF
(hex, non 0)
PA Profile 0
(hex, non 0)
PA Profile 1
(hex, non 0)
PA Profile 2
(hex, non 0)
PA Profile 3
(hex, non 0)
Country Code
(hex, non 0)
Upload Parameters
Escape Manufacturing
Send Termination
Packet Data
Timeout
2
(sec: 1-60)
512 bytes
Flash Destination
Area 1
Browse
Firmware File
STDO.afx.bin
Disconnected
Operation History
10.02 14:22:22: Packet 198 sent with 512 bytes
10.02 14:22:22: Packet 199 sent with 512 bytes
10.02 14:22:22: Packet 200 sent with 512 bytes
10.02 14:22:22: Packet 201 sent with 512 bytes
10.02 14:22:22: Packet 202 sent with 512 bytes
10.02 14:22:22: Packet 203 sent with 512 bytes
10.02 14:22:22: Packet 204 sent with 512 bytes
10.02 14:22:22: Packet 205 sent with 512 bytes
10.02 14:22:22: Packet 206 sent with 512 bytes
10.02 14:22:22: Packet 207 sent with 512 bytes
10.02 14:22:23: Packet 208 sent with 512 bytes
10.02 14:22:23: Packet 209 sent with 512 bytes
10.02 14:22:23: Packet 210 sent with 512 bytes
10.02 14:22:23: Packet 211 sent with 512 bytes
10.02 14:22:23: Packet 212 sent with 512 bytes
10.02 14:22:23: Packet 213 sent with 512 bytes
10.02 14:22:23: Packet 214 sent with 512 bytes
10.02 14:22:23: Packet 215 sent with 512 bytes
10.02 14:22:23: Packet 216 sent with 512 bytes
10.02 14:22:23: Packet 217 sent with 512 bytes
10.02 14:22:23: Packet 218 sent with 512 bytes
10.02 14:22:23: Packet 219 sent with 512 bytes
10.02 14:22:23: Packet 220 sent with 512 bytes
10.02 14:22:23: Packet 221 sent with 512 bytes
10.02 14:22:23: Packet 222 sent with 512 bytes
10.02 14:22:23: Packet 223 sent with 512 bytes
10.02 14:22:24: Packet 224 sent with 512 bytes
10.02 14:22:24: Packet 225 sent with 512 bytes
10.02 14:22:24: Packet 226 sent with 512 bytes
10.02 14:22:24: Packet 227 sent with 512 bytes
10.02 14:22:24: Packet 228 sent with 512 bytes
10.02 14:22:24: Packet 229 sent with 512 bytes
10.02 14:22:24: Packet 230 sent with 512 bytes
10.02 14:22:24: Packet 231 sent with 512 bytes
10.02 14:22:24: Packet 232 sent with 512 bytes
10.02 14:22:24: Packet 233 sent with 512 bytes
10.02 14:22:24: Packet 234 sent with 512 bytes
10.02 14:22:24: Packet 235 sent with 32 bytes
Uploaded 119840 bytes into 235 packets from "D:\Docs\Madalina\FWD

6. After successfully loading, the Firmware RS232 Uploader will display the status message **Disconnected**.
7. Open the serial port.
8. Wait about 10seconds and you can see the status messages from bootloader.
9. Wait another 5seconds to launch the application.



Message error status:

SPI_FLASH_ERR	SPI Flash (external flash) is inaccessible or corrupted OR No image was loaded in external flash
PFLASH_ERR	Image cannot be erased or written to the CPU Flash

BAD_IMG_PFLASH_ADD	An image was compiled that did not have the linker file set for the bootloader
IMAG_CRC_BAD	CRC image corrupted
ERROR	Any other problem

6.3 Upgrading the Smart Object stack firmware OTA using NAMT

See The NAMT for upgrading the Smart Object stack OTA from NAMT.

6.4 Upgrading the Smart Object stack firmware OTA using Cloud-based NOC website

See the Cloud-based NOC User Manual for upgrading the Smart Object stack OTA from Cloud-based NOC website.

6.5 Upgrading the Quark Transceiver firmware using Quark web interface

Follow the steps in section "Upgrading the Versa Router™ 1000 Quark Transceiver".

6.6 Upgrading the Quark Transceiver firmware using Quark /admin/ interface

Follow the steps in section "Upgrade Versa Router™ 1000 Quark Transceiver firmware".

6.7 Upgrading the Versa Router™ 1000 Quark software using Quark web interface

Follow the steps in section "Upgrading the Versa Router™ 1000 Quark Software".

6.8 Upgrading the Versa Router™ 1000 Quark software using Quark /admin/ interface

Follow the steps in section "Upgrade Versa Router™ 1000 Quark Software".

6.9 Upgrading the Versa Router™ 1000 Quark software using Cloud-based NOC website

See the Cloud-based NOC User Manual for upgrading the Quark software from Cloud-based NOC website.

6.10 Upgrading the Versa Router™ 1000 Quark website using Quark /admin/ interface

Follow the steps in section "Upgrade the Versa Router™ 1000 Quark Website".

6.11 Upgrading the NAMT

Follow the steps in section "Installing the NAMT".

7 Configuring the Smart Objects Development Kit

7.1 Overview

This section describes the steps necessary for configuring or re-configuring the Nivis GEN3 M1 Smart Object Integration kit. It provides a set of default configuration parameters and details the steps that need to be followed in order to configure the kit parameters such as the **Network ID**, **Join Key**, enable/disable **PANA**/**DTLS**, and other parameters. The section also describes the steps to be taken to connect to the Cloud-Based NOC.

NOTES

The end user is NOT allowed to convert system configured for US to a system configured for JP or the other way around.

The end user is NOT allowed to modify the channels bitmap (the list of channels to use for frequency hopping).

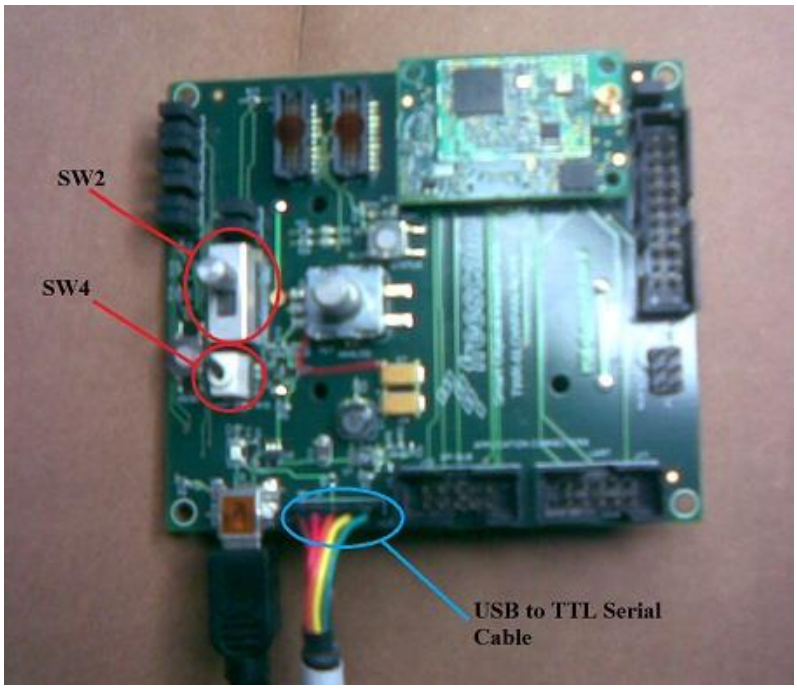
7.2 Accessing the configuration interface on the Smart Object

The configuration interface can be used to change any of the settings on the Smart Object described in the following sections.

Start with a Smart Object that is plugged in the Power Pack board.

Please ensure that the buttons are positioned as shown in the picture below:

- Set switch SW4 in SPI position.
- Set switch SW2 to the top position (SW2 has three positions: 1-toggle, 2-central, 3-top).

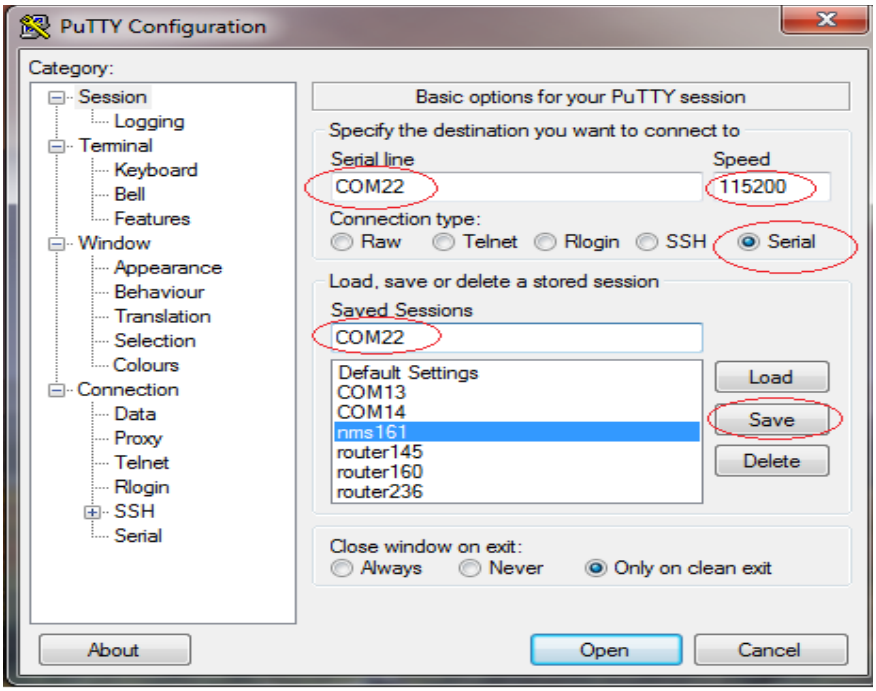


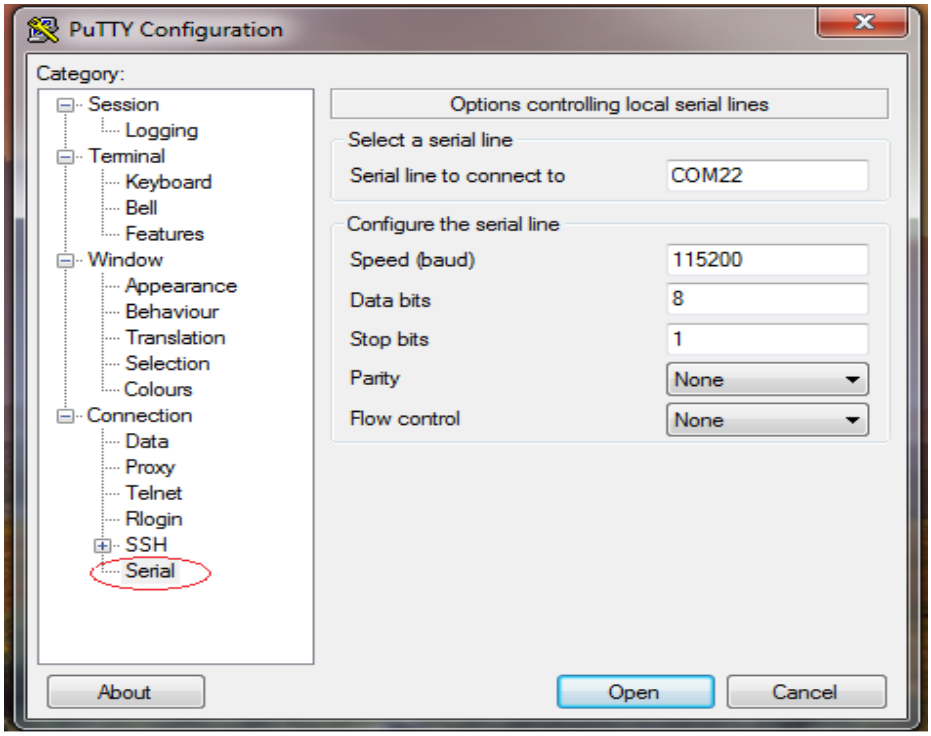
The configuration procedure requires serial connectivity to the Smart Object through connector J5 (labeled "USB Programming Cable") present on the Power Pack board. Connect to the Power Pack board using a USB-to-TTL serial cable with part number TTL-232R-3V3 manufactured by FTDI (manufacturer link: http://www.ftdichip.com/Products/Cables/USBTTLSerial.htm). As shown in the picture, please ensure that the black conductor of the cable is connected to pin 1 (indicated with an arrow) of the J5 connector present on the Power Pack board.

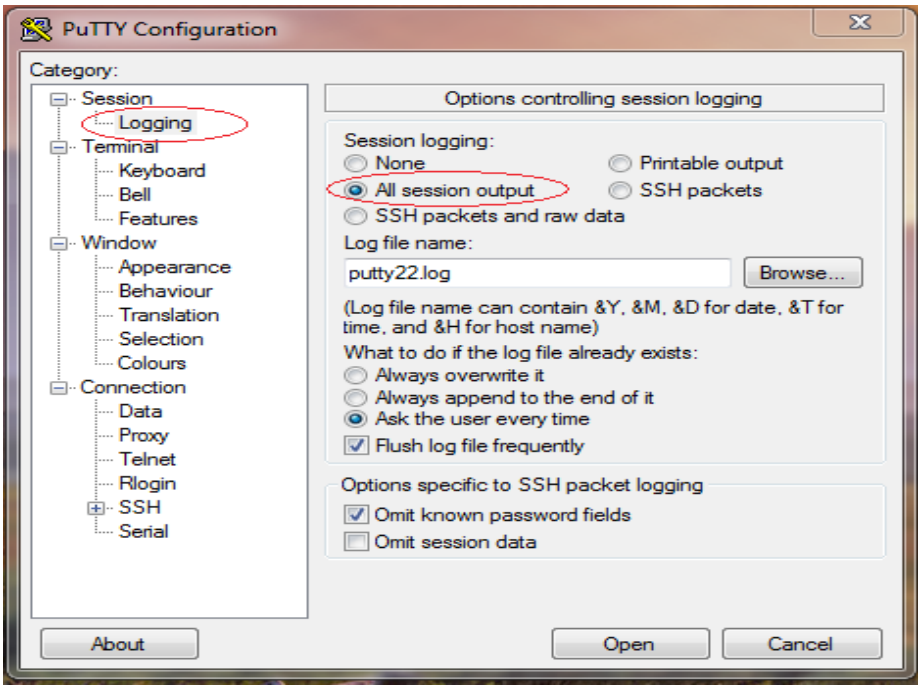
Any serial terminal application can be used. This guide shows how to configure the Smart Object using the PuTTY serial terminal application.

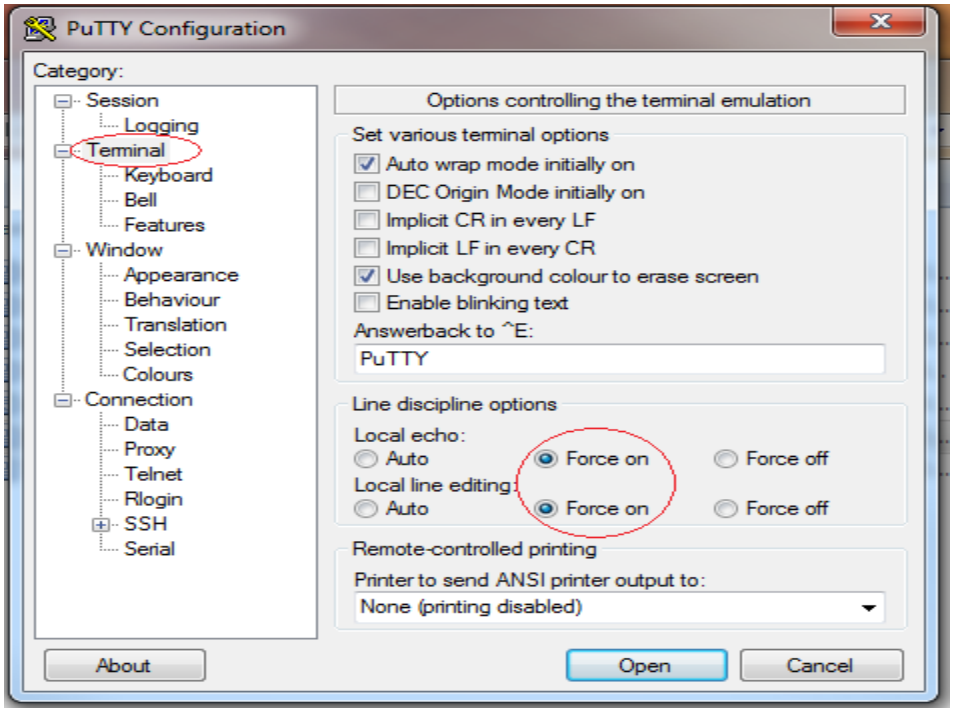
The following section includes a sequence of pictures that indicate how to configure the PuTTY serial terminal.

- Configure the terminal to communicate over the appropriate COM port (COM22 is shown in the figure below as an example)
- Check the **Serial** option.
- Set the **Speed** to 115200.
- Set the COM port.
- Apply various settings in the left panel (see following figures).
- Return to the **Session** panel and click **Save**.


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Serial line
Speed
COM22
115200
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
COM22
Default Settings
COM13
COM14
nms161
router145
router160
router236
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Options controlling local serial lines
Select a serial line
Serial line to connect to
COM22
Configure the serial line
Speed (baud)
115200
Data bits
8
Stop bits
1
Parity
None
Flow control
None
About
Open
Cancel





Once the PuTTY terminal is configured, connect to the Smart Object.

Reset the Smart Object. Once the PuTTY application is connected to the Smart Object, the configuration interface will be available on the PuTTY. Then you can input the configuration commands described in the following sections.

NOTE All settings can be modified in a single session.

7.3 Accessing the configuration interface on the Quark

The Quark website on the Versa Router™ 1000 Quark can be accessed by pointing a web browser to the Quark IP.

See

Configuration and administration of the VersaRouter™ 1000 for details on how to operate the Quark configuration interface.

7.4 Enable/Disable DTLS, PANA and Low Power

7.4.1 Overview

The DTLS settings on the Smart Object must match the corresponding settings on the Quark. Specifically:

- If DTLS is disabled on the Smart Object, the corresponding entry on the Quark must specify cipher suite "DTLS_NOT_USED".
- If DTLS is enabled on the Smart Object, the corresponding entry on the Quark must specify a cipher suite other than "DTLS_NOT_USED".

The PANA is Enabled / Disabled on Smart Object only.

The flags are a bit field with the following meaning:

Bit 0: DTLS enable/disable

Bit 1: PANA enable/disable

Bit 0: Low Power enable/disable (valid on VN400 only)

7.4.2 Enable/Disable low power on Smart Objects – VN400 flavor only

1. Follow the steps described in Section 7.2 "Accessing the configuration interface on the Smart Object".
2. Type ***+dbgon***
3. Type ***+setflags 04***

This command will activate low power, on VN400 flavor only (this exact value will disable DTLS and PANA). The command will have no effect on the Low Power mode on Power Pack flavor.

7.4.3 Enable/Disable DTLS and PANA on the Smart Objects

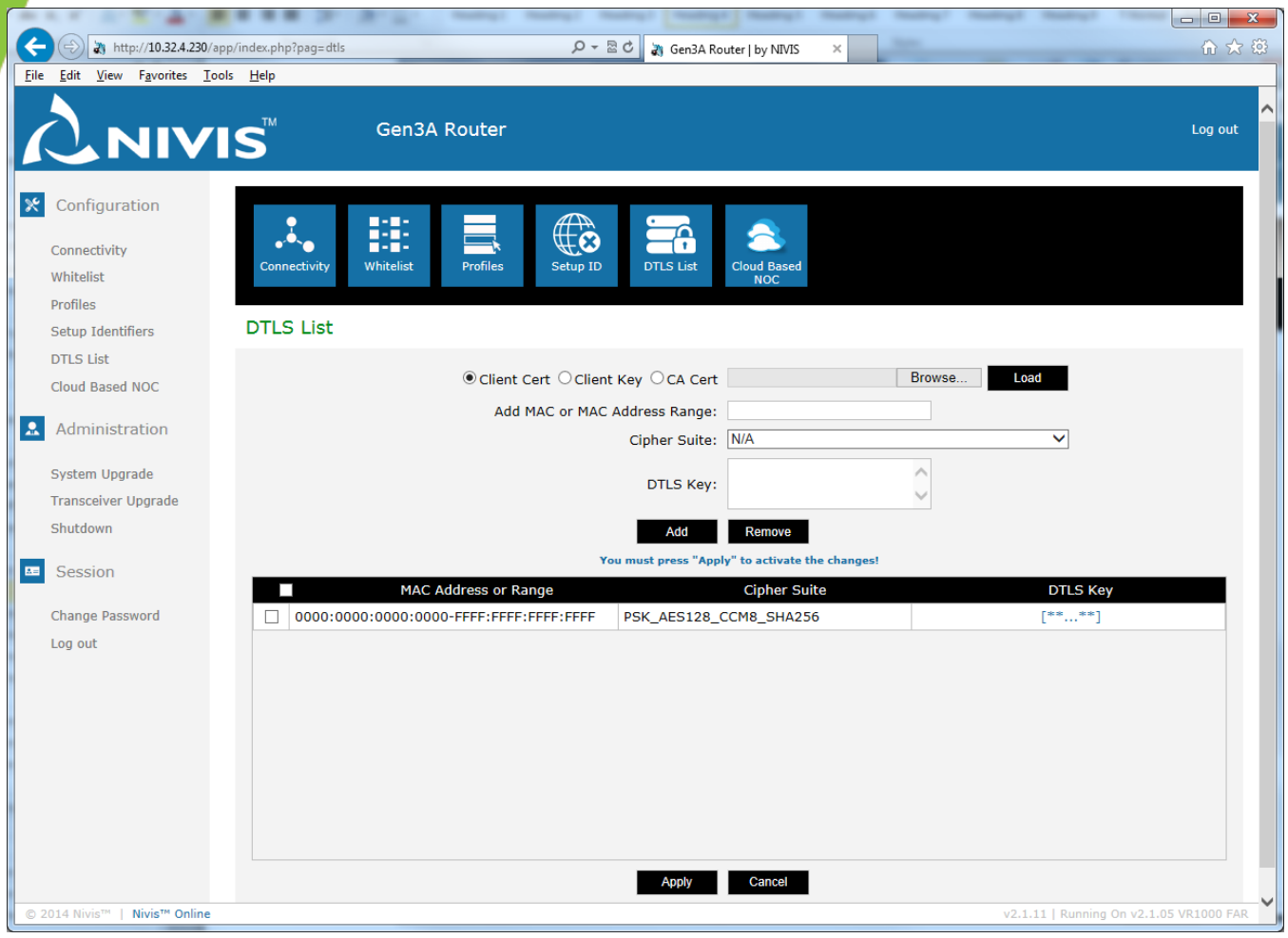
1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type ***+setflags 03*** (this command will activate both *DTLS* and *PANA*). To disable DTLS or PANA or both, use the following flag values:
    - **00**->PANA disabled, DTLS disabled;
    - **01**->PANA disabled, DTLS enabled;
    - **02**->PANA enabled, DTLS disabled;
    - **03**->PANA enabled, DTLS enabled;

The flags can combine with the Low Power bit.

4. Type ***+getflags*** (this command will display current flags of DTLS and PANA). The flag values are described in step 3.

7.4.4 Enable/disable DTLS on the Quark

Click on **DTLS List** in the **Configuration** section on the Quark website to access the DTLS configuration page.



Select the Smart Object MAC, or select a range including the Smart Object MAC, or, if this is a device missing from the list, add the MAC. Then choose the cipher suite, enter the DTLS key if necessary, click **Add**, then click **Apply**.

NOTES

If you are configuring more than a single device, please update the settings for all of the devices, and then click **Apply**.

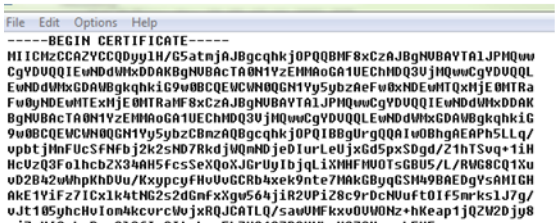
If you choose to enable DTLS and choose a cipher suite that requires a key or certificate, you will need to choose a matching key/certificate on the Smart Object and Quark; see Change the DTLS Key/certificate on the Smart Object and

Change the DTLS Key/Certificate on the Quark for details.

See Configuring the Versa Router™ 1000 Quark DTLS list for details on setting the DTLS on the Quark.

7.4.5 The Smart Objects certificates

There Smart Object uses the same type of certificates for DTLS and for PANA. A certificate for the Smart Object consists of 3 files: Private Key, Public Key and Certification Authorities (CA).



7.4.5.1 Processing certificates

In order to write to the device they must be initially processed in a format recognized by the device. You need to copy text (between -----BEGIN ……. And -----END… ----- in a text editor (example: notepad++) and replacing all [\r\n] with [*]).This will look like below.

```
-----BEGIN CERTIFICATE-----*MIICMzCCAZYC.....9VqnmbFHE*-----END CERTIFICATE-----*
```

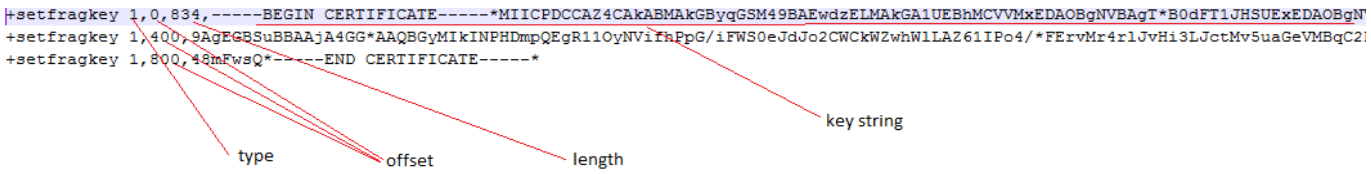
All the strings have to has the maximum length: APPSEC_MAX_PSK_SIZE [64], APPSEC_MAX_PUB_CERT_SIZE [850] (public key), APPSEC_MAX_PRI_KEY_SIZE [462] (private key), APPSEC_MAX_CERT_AUTH_SIZE [1280] (certificate authority);

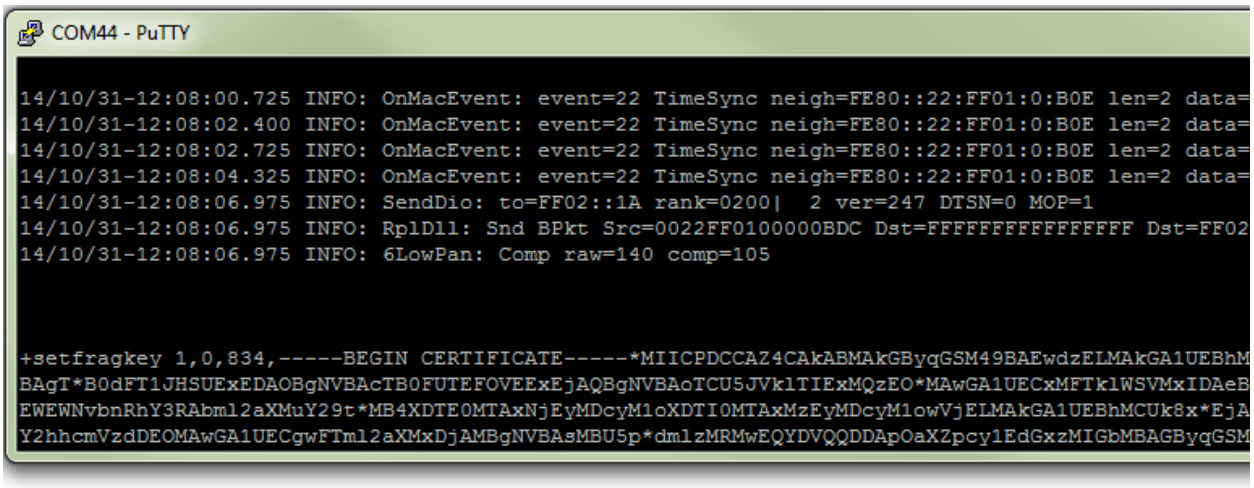
7.4.5.2 Writing certificates into Smart Object

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type +setfragkey type,off,[len,]key

Where:

- type
  - *APPSEC_KEY_DTLS_PSK* [0],
  - *APPSEC_KEY_DTLS_PUB_CERT* *[1]*,
  - *APPSEC_KEY_DTLS_PRI_KEY* *[2]*,
  - *APPSEC_KEY_DTLS_CA* *[3]*,
  - *APPSEC_KEY_PANA_PUB_CERT* *[4]*,
  - *APPSEC_KEY_PANA_PRI_KEY* *[5]*,
  - *APPSEC_KEY_PANA_CA* *[6]*,
- off – offset in file
- len – length of the file. This appears in the first command. See at example below.
- key – the processing string (maximum 450 characters).

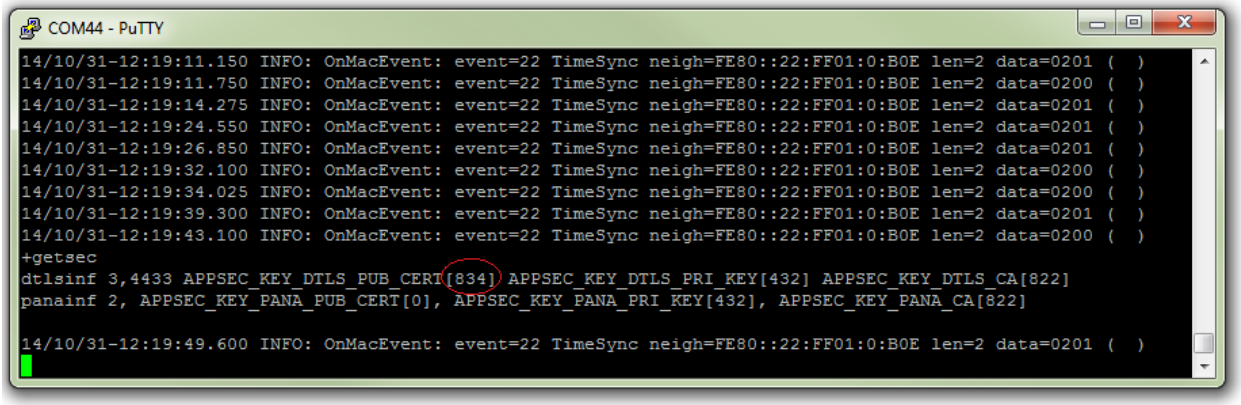



COM44 - PuTTY
14/10/31-12:08:00.725 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=
14/10/31-12:08:02.400 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=
14/10/31-12:08:02.725 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=
14/10/31-12:08:04.325 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=
14/10/31-12:08:06.975 INFO: SendDio: to=FF02::1A rank=0200| 2 ver=247 DTSN=0 MOP=1
14/10/31-12:08:06.975 INFO: RplDll: Snd BPkt Src=0022FF0100000BDC Dst=FFFFFFFFFFFFFFFF Dst=FF02
14/10/31-12:08:06.975 INFO: 6LowPan: Comp raw=140 comp=105
+setfragkey 1,0,834,-----BEGIN CERTIFICATE-----*MIICPDCCAZ4CAkABMAkGByqGSM49BAEwdzELMAkGA1UEBhM
BAgT*B0dFT1JHSUExEDAOBgNVBAcTB0FUTEFOVEExEjAQBgNVBAoTCU5JVklTIExMQzEO*MAwGA1UECxMFTklWSVMxIDAeB
EWEWNvbnRhY3RAbml2aXMuY29t*MB4XDTE0MTAxNjEyMDcyM1oXDTI0MTAxMzEyMDcyM1owVjELMAkGA1UEBhMCUk8x*EjA
Y2hhcmVzdDEOMAwGA1UECgwFTml2aXMxDjAMBgNVBAsMBU5p*dmlzMRMwEQYDVQQDDApOaXZpcy1EdGxzMIGbMBAGByqGSM

7.4.5.3 *Verify Smart Object certificates*

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type +getsec

The security information will appear as shown below:


COM44 - PuTTY
14/10/31-12:19:11.150 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )
14/10/31-12:19:11.750 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0200 ( )
14/10/31-12:19:14.275 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )
14/10/31-12:19:24.550 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )
14/10/31-12:19:26.850 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )
14/10/31-12:19:32.100 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0200 ( )
14/10/31-12:19:34.025 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0200 ( )
14/10/31-12:19:39.300 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )
14/10/31-12:19:43.100 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0200 ( )
+getsec
dtlsinf 3,4433 APPSEC_KEY_DTLS_PUB_CERT[834] APPSEC_KEY_DTLS_PRI_KEY[432] APPSEC_KEY_DTLS_CA[822]
panainf 2, APPSEC_KEY_PANA_PUB_CERT[0], APPSEC_KEY_PANA_PRI_KEY[432], APPSEC_KEY_PANA_CA[822]
14/10/31-12:19:49.600 INFO: OnMacEvent: event=22 TimeSync neigh=FE80::22:FF01:0:B0E len=2 data=0201 ( )

7.5 Change the DTLS Certificate / pre-shared Key

7.5.1 Overview

When DTLS is enabled and the cipher suite chosen require a pre-shared Key or a Certificate, the Key or Certificate on the Smart Object and Quark must match.

See Configuring the Versa Router™ 1000 Quark DTLS list for details on which cipher suites require a pre-shared Key and which cipher suites require a Certificate.

7.5.2 Change the DTLS Key/certificate on the Smart Object

7.5.2.1 Choose the DTLS type on the Smart Object

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type + setdtlsinf <type>,<port>

Where:

<type> is DTLS_TYPE_PSK [1], DTLS_TYPE_CERT [2], DTLS_TYPE_PSK_AND_CERT [3]

<port> is the DTLS port (4433)

7.5.2.2 Change the DTLS certificate on the Smart Object

The DTLS certificate is used only if the DTLS type is selected as DTLS_TYPE_CERT [2] or DTLS_TYPE_PSK_AND_CERT [3].

Follow the steps described in Section “Writing certificates into Smart Object”

Use for type on of the following:

a. *APPSEC_KEY_DTLS_PSK* [0],
b. *APPSEC_KEY_DTLS_PUB_CERT [1],*
c. *APPSEC_KEY_DTLS_PRI_KEY [2],*
d. *APPSEC_KEY_DTLS_CA [3],*

7.5.2.3 Change the DTLS Key on the Smart Object

The DTLS key is used only if the DTLS type is selected as DTLS_TYPE_PSK [1] or DTLS_TYPE_PSK_AND_CERT [3].

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type +setdtlskey <type>,<len>,<key>

Where:

<type> is 0 for DTLS_KEY_PSK; or 1 for DTLS_KEY_CERT;

<len> is the key length

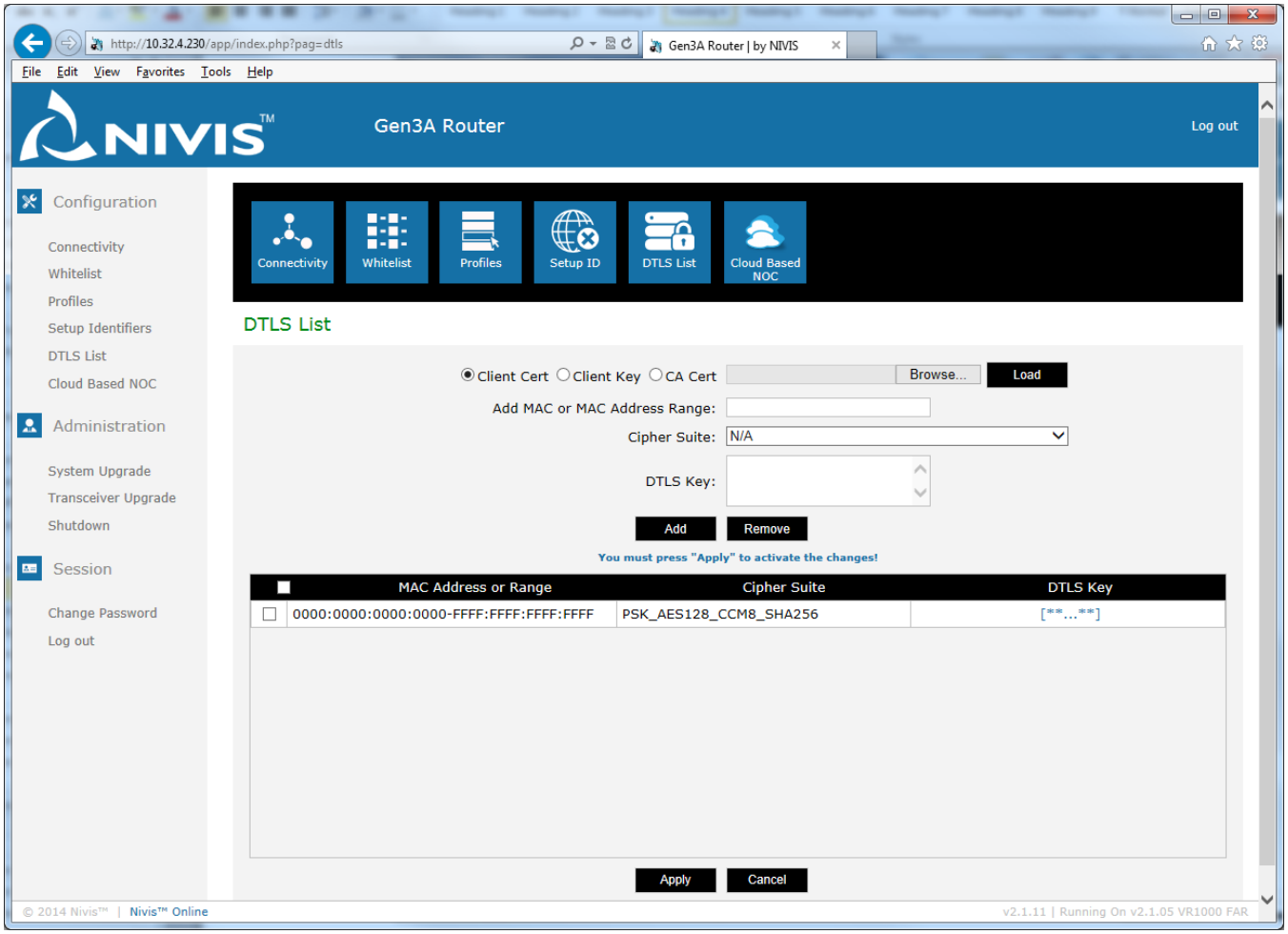
<key> is PSK or CERT in hex

Example:

+setdtlskey 0,4,1A2B3C4D

7.5.3 Change the DTLS Key/Certificate on the Quark

Click on **DTLS List** in the **Configuration** section on the Quark website to access the DTLS configuration page.



7.5.3.1 Change the DTLS Certificate on the Quark

Select the file type to load (Client Certificate/Client key/CA Certificate), browse for the file, and click **Load**.

NOTES
All three file types must be loaded on the Quark for a Certificate-based cipher suite to work.
Currently the Quark supports a single certificate per network.

7.5.3.2 Change the DTLS Key on the Quark

Select the Smart Object MAC, or select a range including the Smart Object MAC, or, if this is a device missing from the list, add the MAC. Then choose the cipher suite (a suite using a pre-shared Key), enter the DTLS key, click **Add**, then click **Apply**.

See Configuring the Versa Router™ 1000 Quark DTLS list for details on setting the DTLS on the Quark.

7.6 Change the PANA Certificate / pre-shared Join Key

Network admission can be done based on pre-shared keys or based on PANA Certificates. The two methods are mutually exclusive.

7.6.1 Choose the PANA type on the Smart Object

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type ***+setpanainftype <type>***

Where type is: PANA_TYPE_PSK [1], PANA_TYPE_CERT [2]

The newly configured PANA type will be used the next time the Smart Objects joins the network.

NOTES
In order to force a device rejoin without power cycling the device, the user can use the command: +setrejoin timeout,3

7.6.2 Change the PANA Certificate

7.6.2.1 Change the PANA certificate on the Smart Object

The PANA certificate is only used if the PANA type is selected as PANA_TYPE_CERT [2].

Follow the steps described in Section Writing certificates into Smart Object

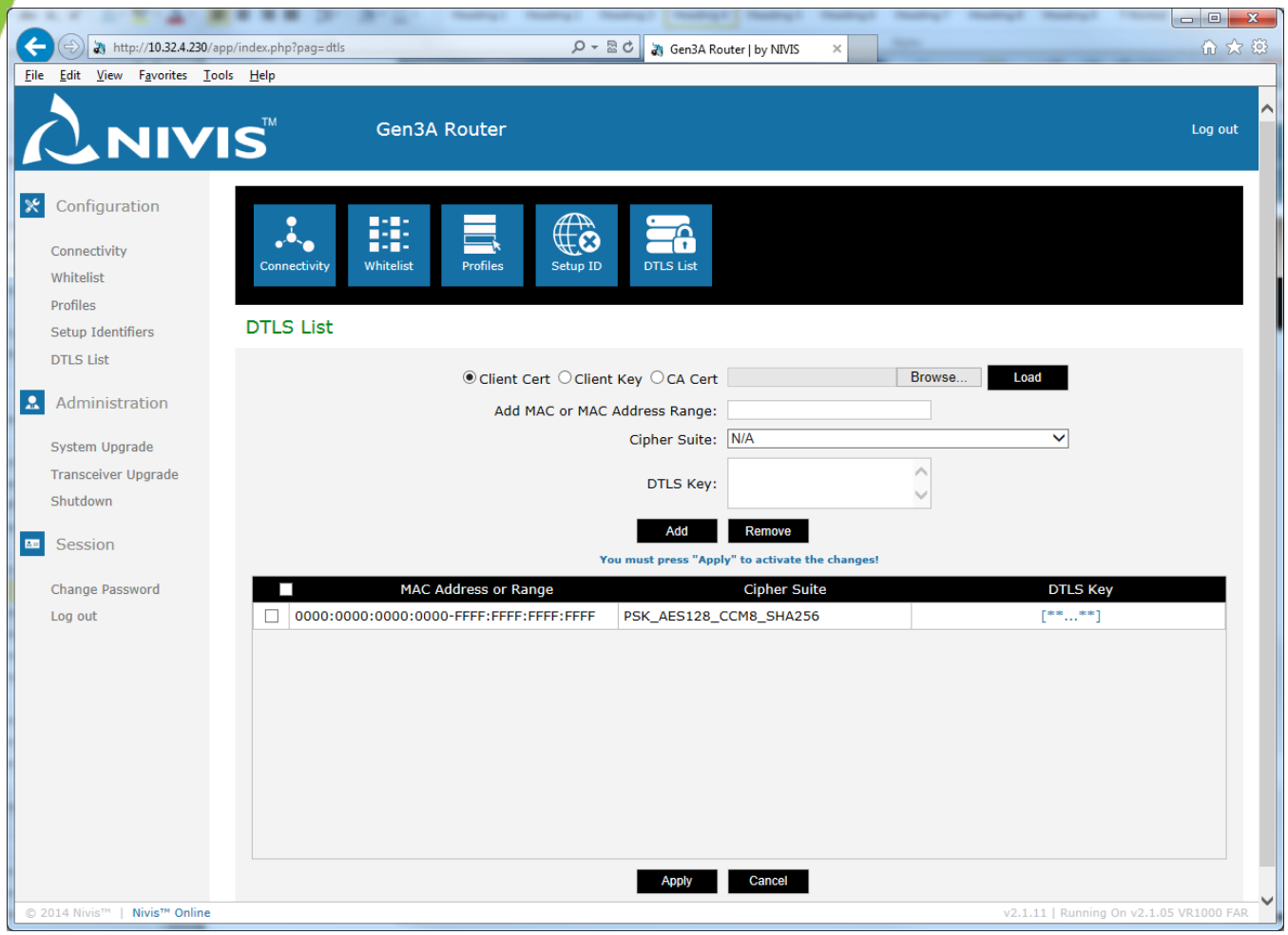
Use for type

- *APPSEC_KEY_PANA_PUB_CERT [4],*
- *APPSEC_KEY_PANA_PRI_KEY [5],*
- *APPSEC_KEY_PANA_CA [6],*

NOTES
All three types must be loaded on the Smart Object for a Certificate-based cipher suite to work.

7.6.2.2 *Change the PANA Certificate on the Quark*

Click on **DTLS List** in the **Configuration** section on the Quark website to access the DTLS configuration page. The PANA certificates are loaded using the same page as for DTLS.



Select the file type to load (Client Certificate/Client key/CA Certificate), browse for the file, and click **Load**.

NOTES

All three file types must be loaded on the Quark for a Certificate-based cipher suite to work.

Currently the Quark supports a single certificate per network.

7.6.3 Change the pre-shared Join Key

7.6.3.1 Change the pre-shared Join Key on the Smart Object

1. Follow the steps described in Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type +setsec <minseclevel>,<crtseclevel>,<keyexchange>,<key>

where:

<minseclevel> is the minimum security level

<crtseclevel> is the current security level

<keyexchange> is 1 for key exchange, and 0 for NO key exchange (pre-shared key)

<key> is the key in hex

Example (PANA Disabled):

+setsec 1,7,0,5365637265744b657900000000000000

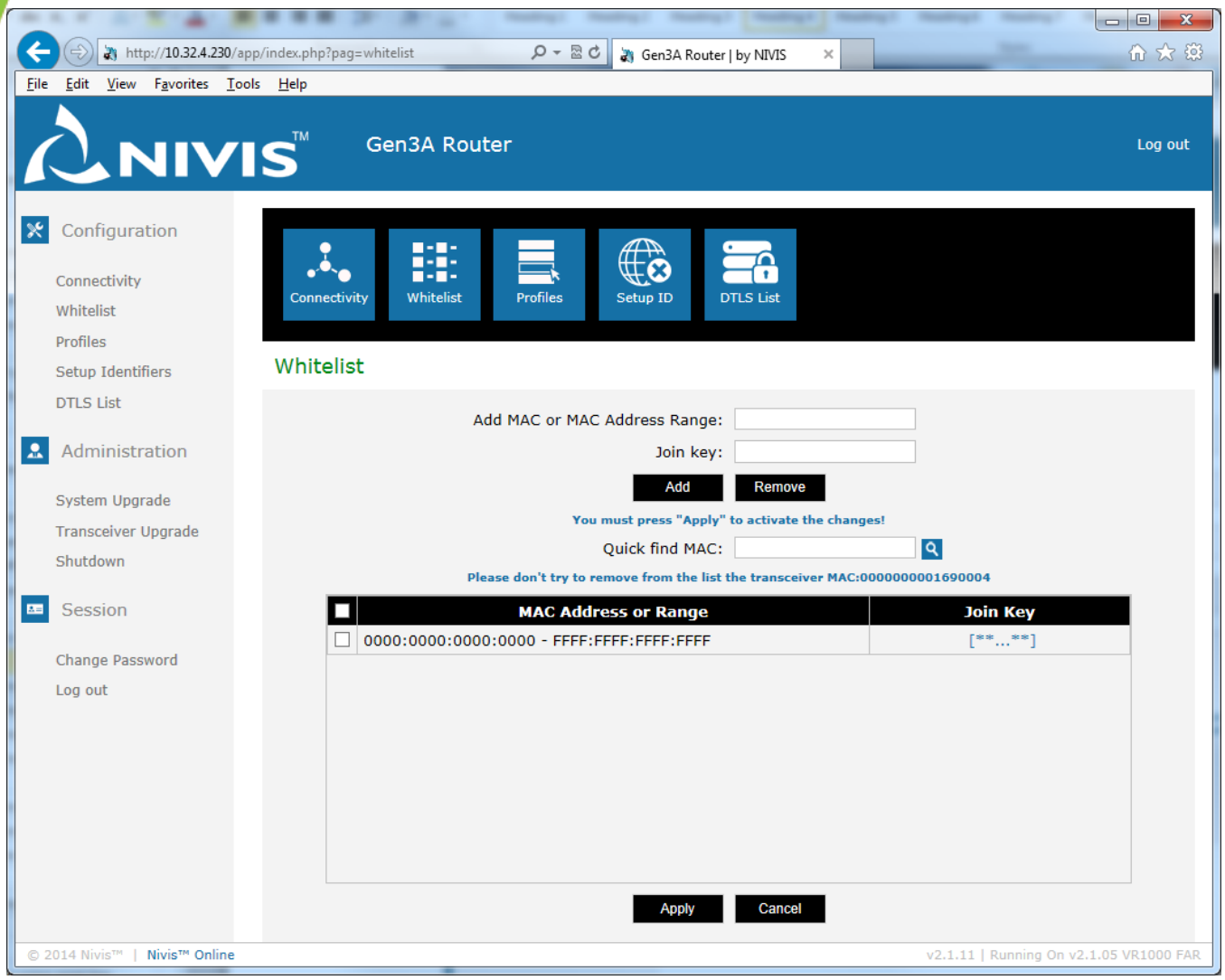
Example (PANA Enabled):

+setsec 1,7,1,5365637265744b657900000000000000

7.6.3.2 *Change the pre-shared Join Key on the Quark*

The instructions below are valid for both PANA Enabled and Disabled.

Click on **Whitelist** in the **Configuration** section on the Quark website.



Select the Smart Object MAC, or select a range including the Smart Object MAC, or, if this is a device missing from the list, add the MAC. Then enter the Join Key, click **Add**, then click **Apply**.

See Configuring the Versa Router™ 1000 Quark Whitelist for details on setting the Join Key on the Quark.

7.7 Change the Network ID

7.7.1 Overview

The Net ID on the Smart Object must match the Vendor Network ID on the Quark to which it needs to join.

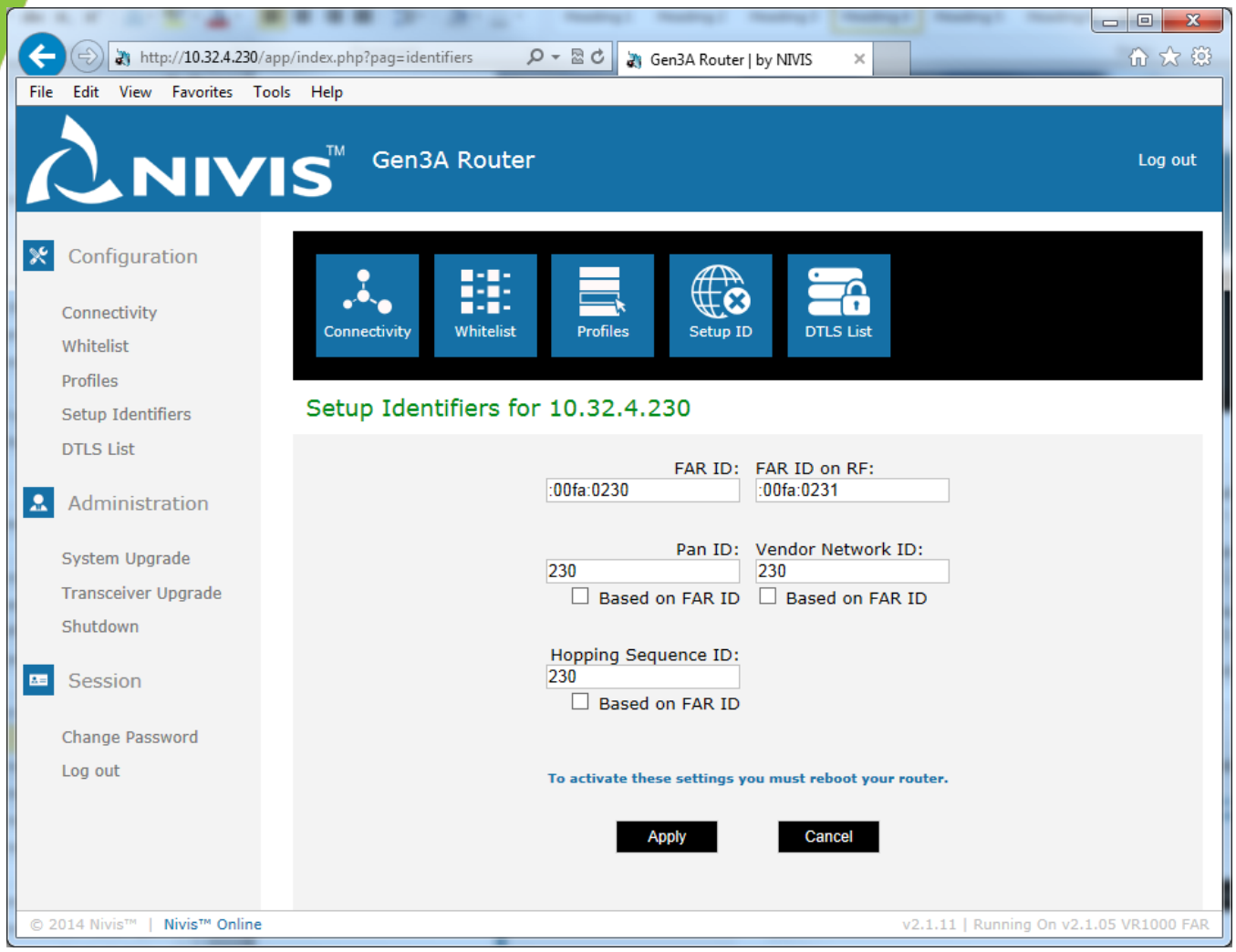
NOTE With default settings, there is NO need to change the Network ID on the Smart Objects or the Quark.

7.7.2 Change the Net ID on the Smart Object

1. Follow the steps described In Section 7.2 “Accessing the configuration interface on the Smart Object”.
2. Type ***+dbgon***
3. Type ***+setnetid 101*** (this will set the subnet id to 101).

7.7.3 Change the Vendor Network ID on the Quark

Click on **Setup Identifiers** in the **Configuration** section on the Quark website.



Modify the **Vendor Network ID**, then click **Apply**.

NOTE

Please be aware, the Quark will reboot and the network will unjoin. The Smart Objects will rejoin only after their Net ID settings match the Vendor Network ID on the Router.

See Configuring the Versa Router™ 1000 Quark Identifiers for details on setting the Vendor Network ID on the Quark.

7.8 Set-up Quark / Cloud-Based NOC communication

Normal deployment scenario requires that the NMS - a subsystem of the Cloud-Based NOC – expose a public IPv4 address. The Quark may or may not have a public IPv4 address.

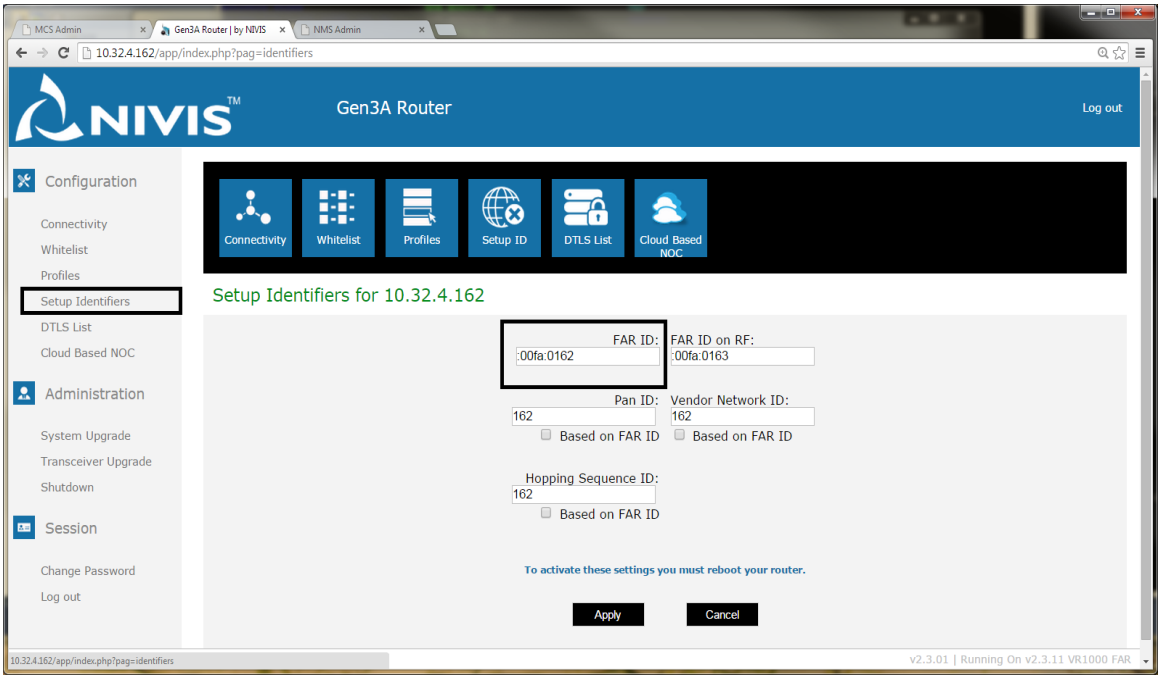
Recommended configuration is to use VPN for Quark-NMS connectivity.

Non-VPN communication is only acceptable for laboratory tests.

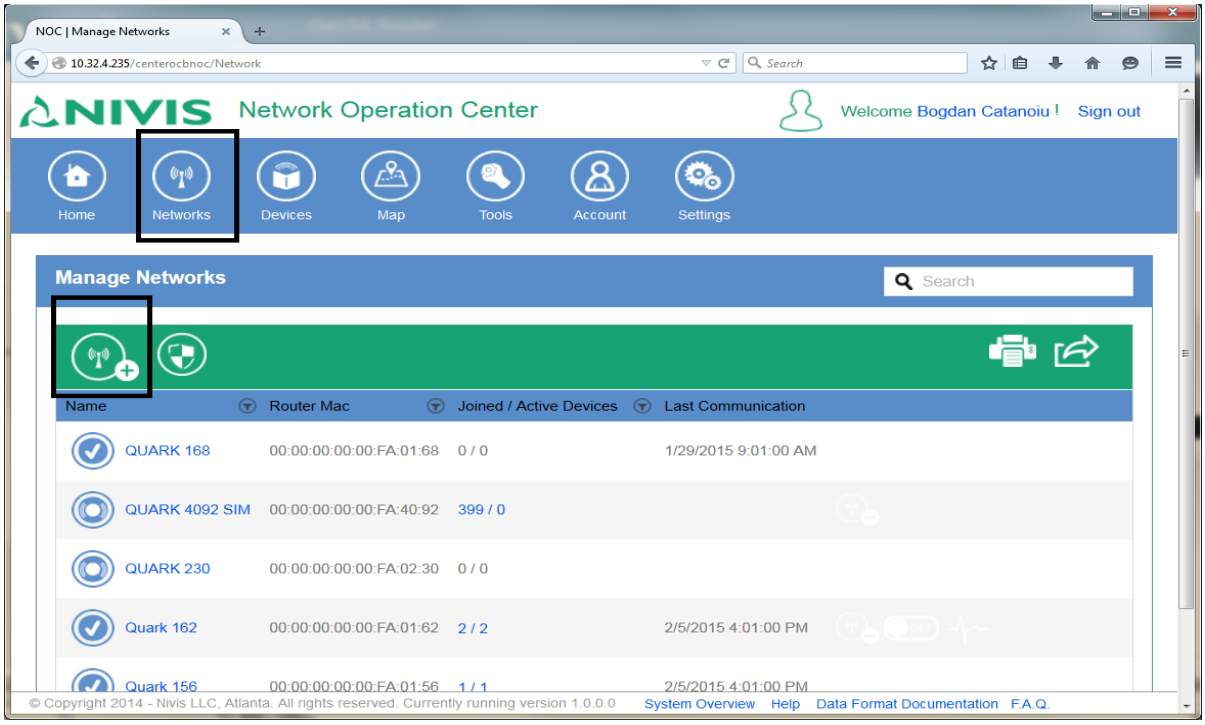
7.8.1 Add Quark in the Cloud-Based NOC Whitelist

Before trying to connect a Quark to the Cloud-Based NOC, the Quark FAR ID must be added to the Cloud-based NOC. The Cloud-Based NOC identifies networks using their controlling Quark.

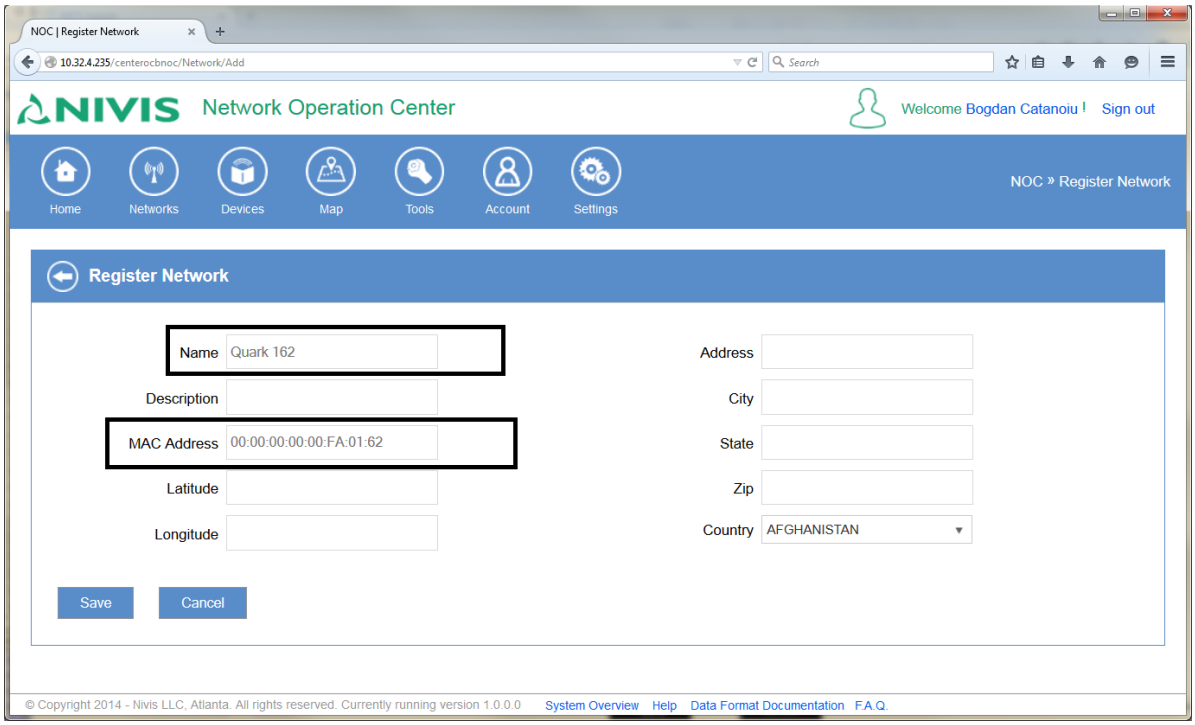
15. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
16. Go to “Setup Identifiers” page
17. Read or copy the FAR ID



18. Point a web browser to the Cloud-Based NOC website, enter credentials
19. Go to on “Networks” page
20. Press “Register Network” button


NOC | Manage Networks
10.32.4.235/centerocbnoc/Network
NIVIS Network Operation Center
Welcome Bogdan Catanoiu ! Sign out
Home
Networks
Devices
Map
Tools
Account
Settings
Manage Networks
Search
Name
Router Mac
Joined / Active Devices
Last Communication
QUARK 168 00:00:00:00:00:FA:01:68 0 / 0 1/29/2015 9:01:00 AM
QUARK 4092 SIM 00:00:00:00:00:FA:40:92 399 / 0
QUARK 230 00:00:00:00:00:FA:02:30 0 / 0
Quark 162 00:00:00:00:00:FA:01:62 2 / 2 2/5/2015 4:01:00 PM
Quark 156 00:00:00:00:00:FA:01:56 1 / 1 2/5/2015 4:01:00 PM
© Copyright 2014 - Nivis LLC, Atlanta. All rights reserved. Currently running version 1.0.0.0
System Overview Help Data Format Documentation F.A.Q.

21. Provide a network Name.
22. Use the Quark FAR ID identified at step 17 above to provide the “MAC Address”.
23. Press “Save”.



7.8.2 Set-up Quark and NMS communication using VPN [RECOMMENDED]

These steps below load Open VPN certificates into NMS and Quark. The steps are performed **only once** for NMS, and **only once** for each Quark, and then the Quark can be switched between "transparent" and "standalone" modes indefinitely without the need to re-load certificates.

Only if the root CA was changed on the Cloud-Based NOC (or when the Quark needs to connect to a different cloud-Based NOC, with a different root CA) the Quark will need to load a new certificate – generated using the new root CA.

NOTE

The certificate sets for Quark and NMS must be delivered in **.tar.gz** format.

The Quark certificate set is specific per Quark.

Do NOT attempt to use a certificate set built for a Quark on a different Quark.

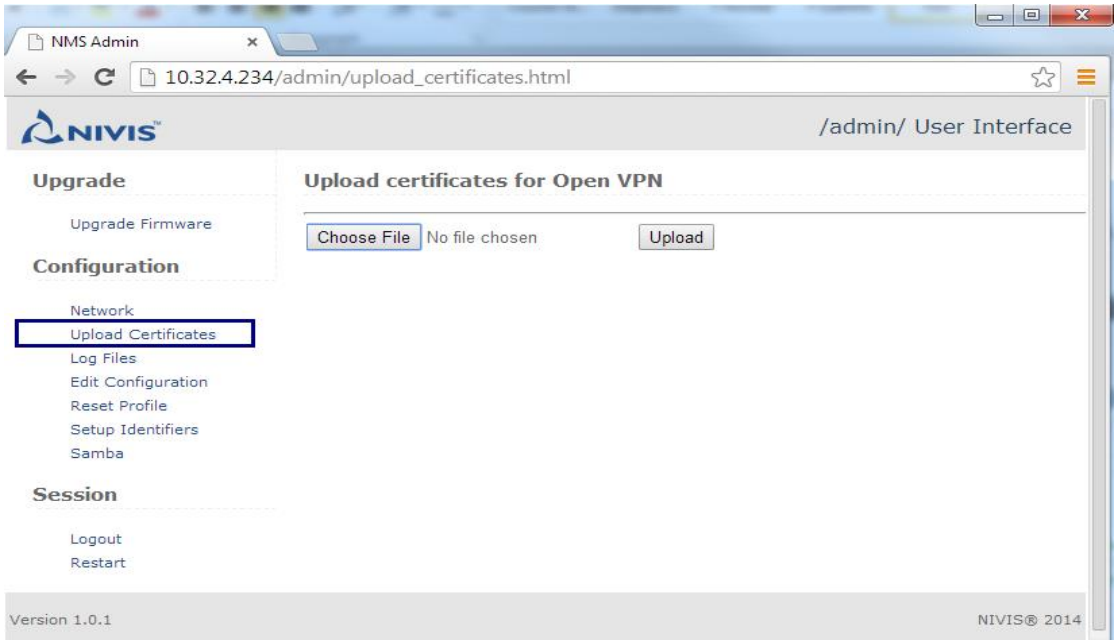
The GEN3 M1 Quarks are delivered with a set of certificates pre-installed. The certificate set is compatible only with Cloud-based NOC hosted by Nivis.

There's no need to upload new certificate set on GEN3 M1 Quarks connecting to Nivis-hosted Cloud-based NOC.

In any other case, the certificate set needs to be uploaded to the Quark

7.8.2.1 *Set-up NMS VPN certificates*

24. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
25. Go to “Upload Certificates” page
26. Press “Choose File” button
27. Select the archive holding the NMS certificate set
28. Press “Upload” button

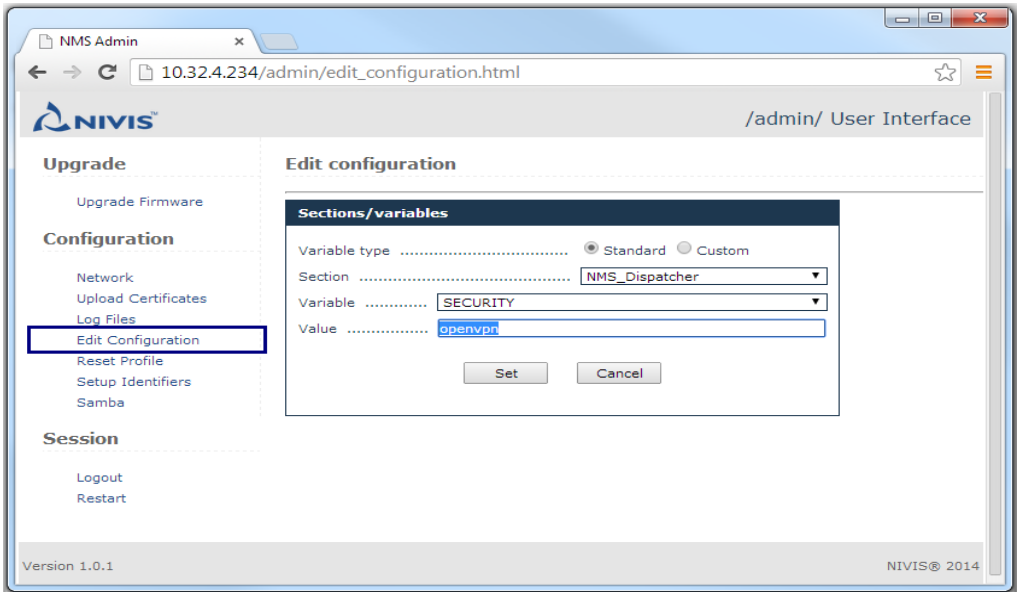


NOTE

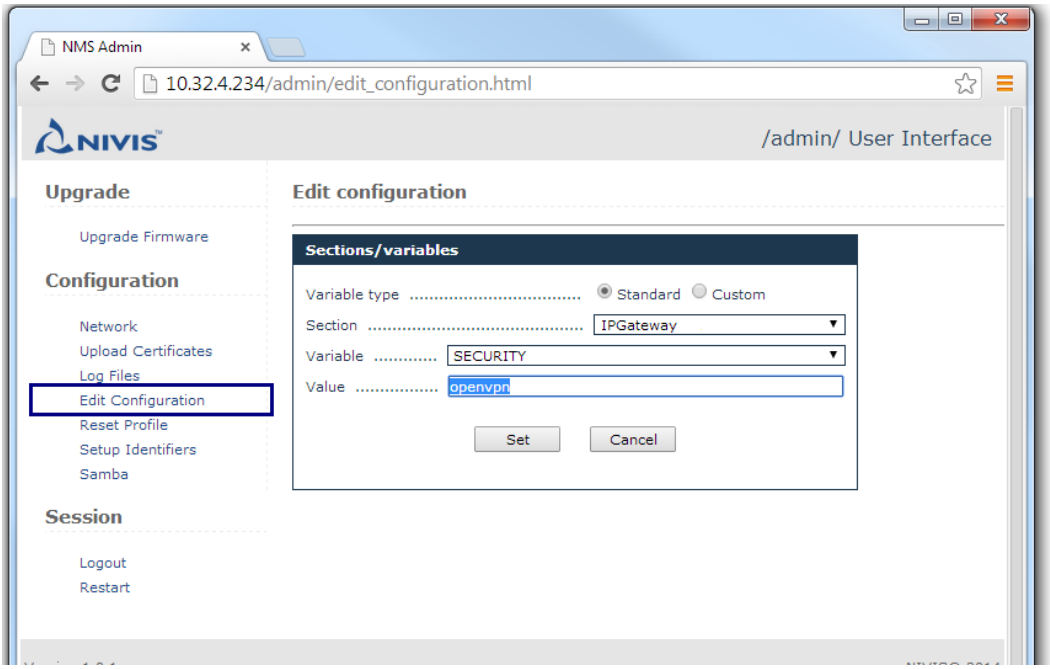
This steps has to be done only once, regardless of the number of subsequent switches of communication between “openvpn” and “none”

7.8.2.2 *Configure NMS modules to use OpenVPN*

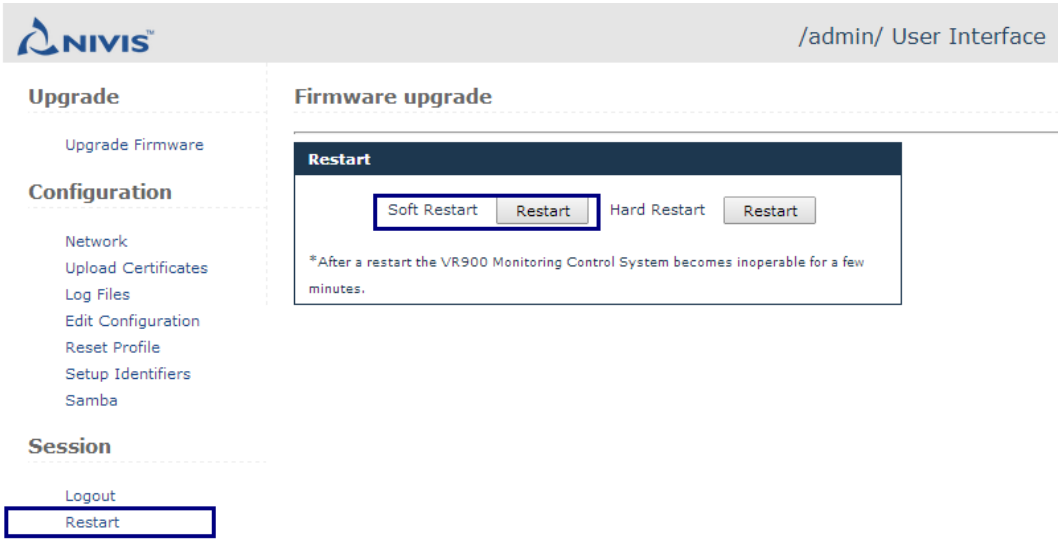
11. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
12. Go to “Edit configuration” page
13. Choose “NMS_Dispatcher” from “Section” drop-down
14. Choose “SECURITY” in “Variable“ drop-down
15. Make sure the “Value” edit reads “openvpn” (no quotes, not other char)



16. Choose “IPGateway” from “Section” drop-down
17. Choose “SECURITY” in “Variable“ drop-down
18. Make sure the “Value” edit reads “openvpn” (no quotes, not other char)

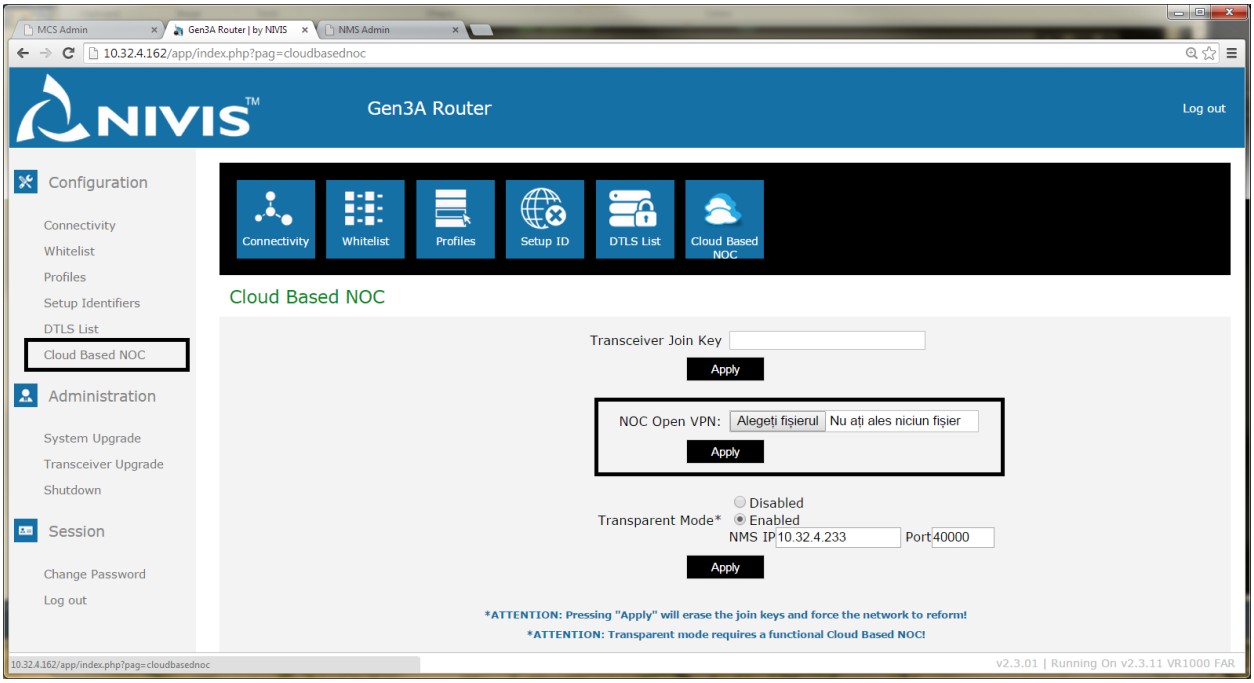


19. Go to “Restart” page
20. Perform a “Soft Restart”



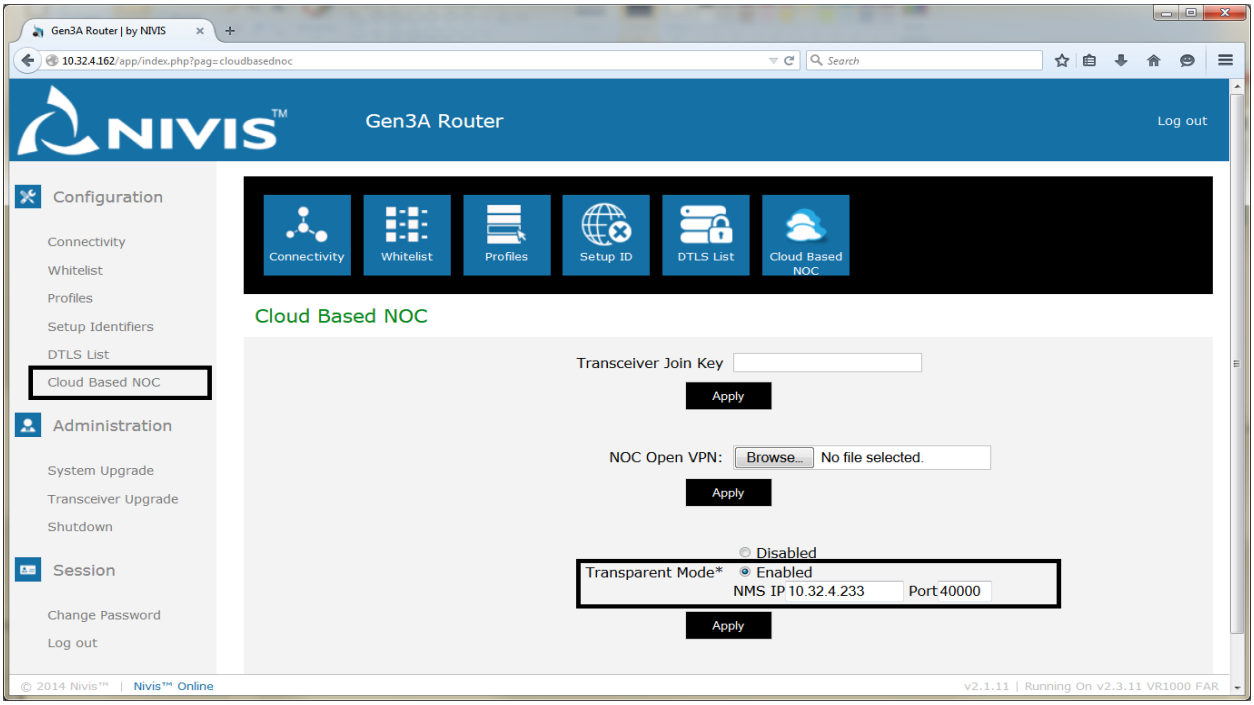
7.8.2.3 *Set-up Quark VPN certificates*

6. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
7. Go to “Cloud Based NOC” page
8. Press the “Choose File” button corresponding to ‘NOC Open VPN’ label
9. Browse for the archive with the Quark certificate set
10. Press the corresponding “Apply” button



7.8.2.4 *Set-up Quark in "transparent" mode (connect to NMS, use openvpn)*

7. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
8. Go to "Cloud Based NOC" page
9. Click to Enable the transparent mode
10. Fill in the NMS **Public IPv4**
11. Fill in the NMS port (default: 40000)
12. Press "Apply" button



NOTE

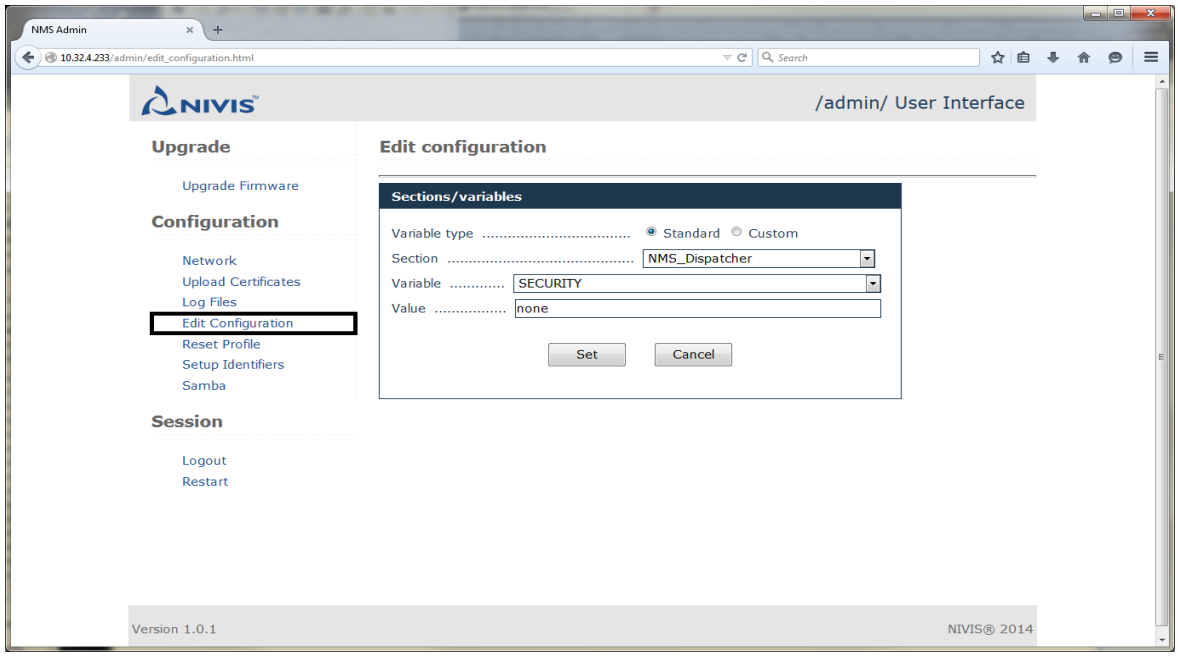
Enabling "transparent" mode on Quark automatically enables the "openvpn" communication for Quark relevant module (BHManager)

7.8.3 [Alternate] Set-up Quark-NMS plain-text communication (Use for lab tests only)

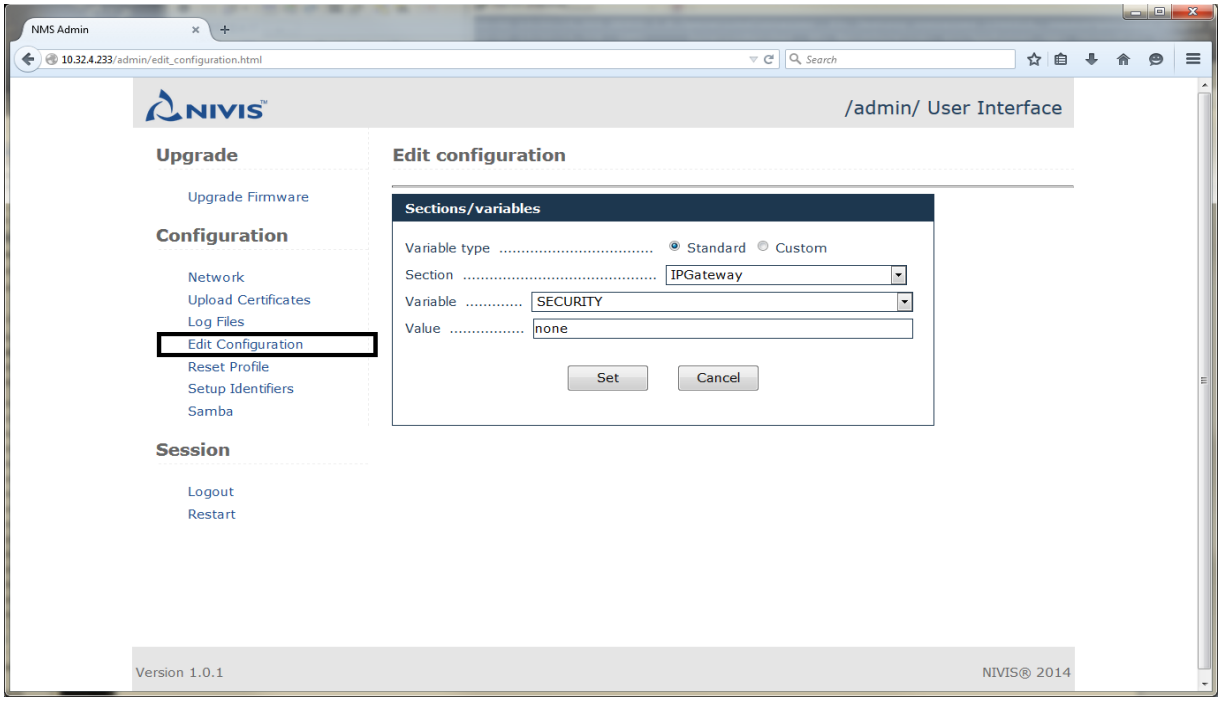
These steps below configure NMS and Quark for non-VPN (open/plaintext) communication. The steps are performed **only once** for NMS, but **each time** the Quark is switched between "standalone" and "transparent" mode, because setting the "transparent" mode automatically enable VPN communication and this step disables it.

7.8.3.1 Configure NMS modules to use open communication

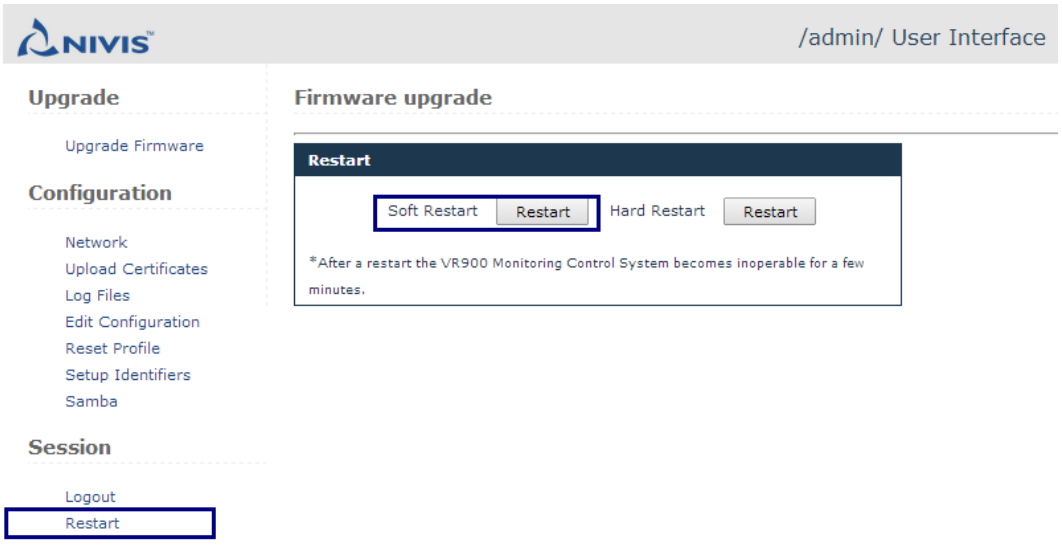
11. Point a web browser to the NMS /admin/ website (http://<nms_ip>/admin/), enter credentials
12. Go to "Edit configuration" page
13. Choose "NMS_Dispatcher" from "Section" drop-down
14. Choose "SECURITY" in "Variable" drop-down
15. Set the field "Value" to "none" (no quotes, not other char)



16. Choose “IPGateway” from “Section” drop-down
17. Choose “SECURITY” in “Variable“ drop-down
18. Set the field “Value” to “none” (no quotes, not other char)

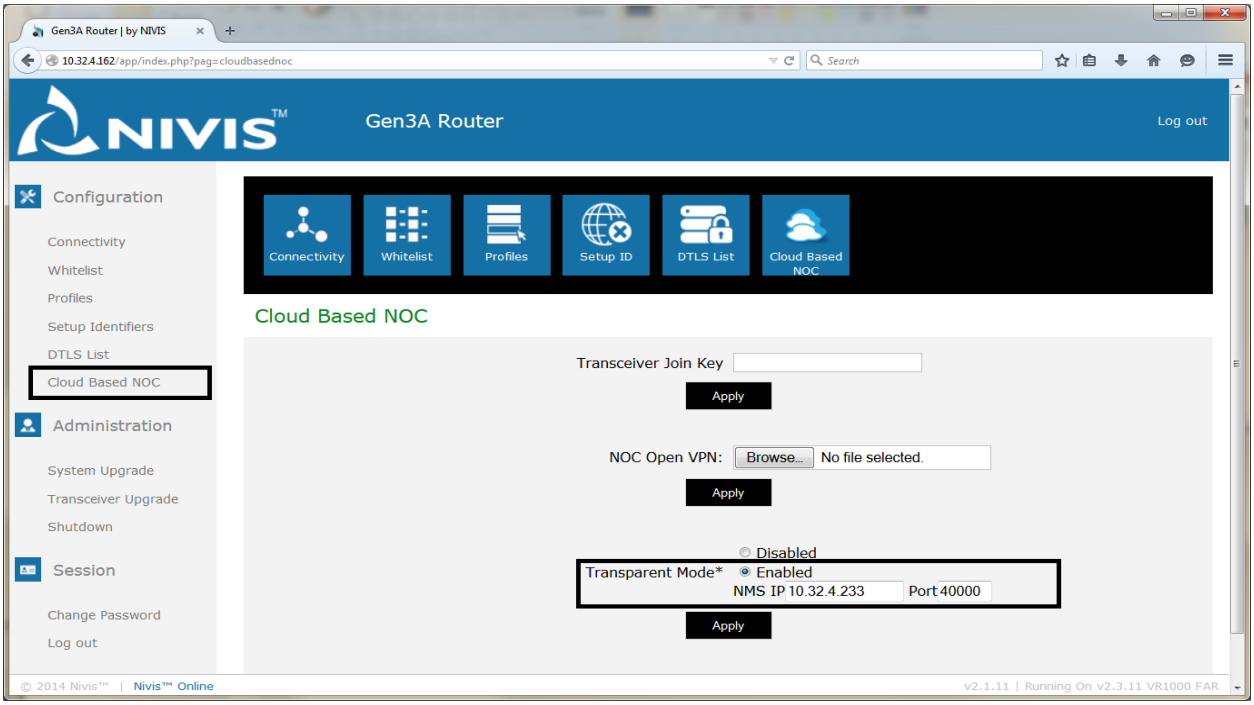


19. Go to “Restart” page
20. Perform a “Soft Restart”



7.8.3.2 *Set-up the Quark in "transparent" mode – connect to the Cloud-based NOC*

7. Point a web browser to the Quark website (http://<quark_ip>/), enter credentials
8. Go to "Cloud Based NOC" page
9. Click to Enable the transparent mode
10. Fill in the NMS **Public IPv4**
11. Fill in the NMS port (default: 40000)
12. Press "Apply" button



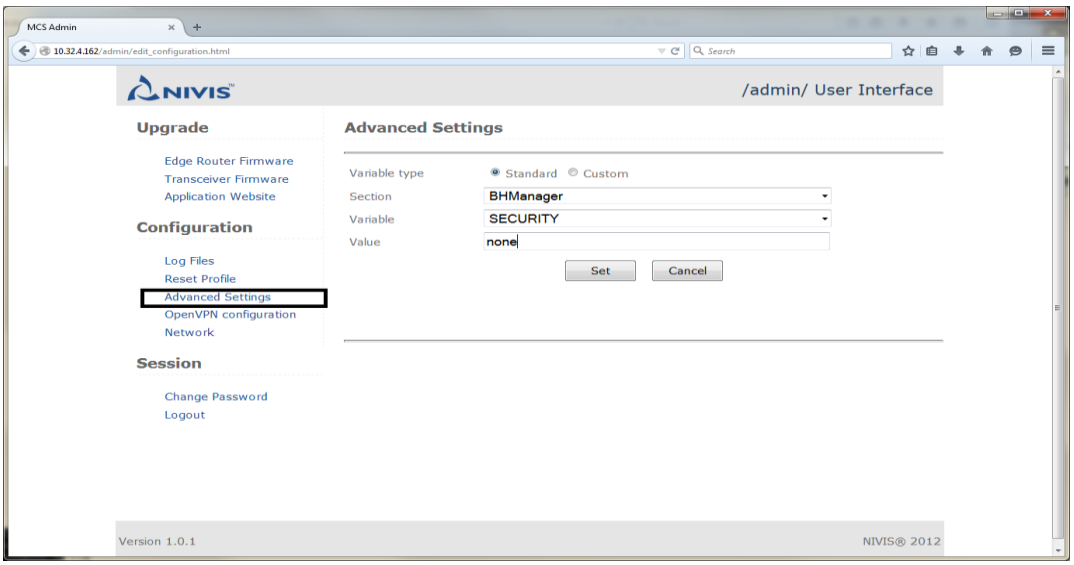
NOTE

Enabling "transparent" mode on Quark automatically enables the "openvpn" communication for Quark relevant module (BHManager). Sine this step uses open communication, the next step is necessary to set the communication back to "open" for BHManager

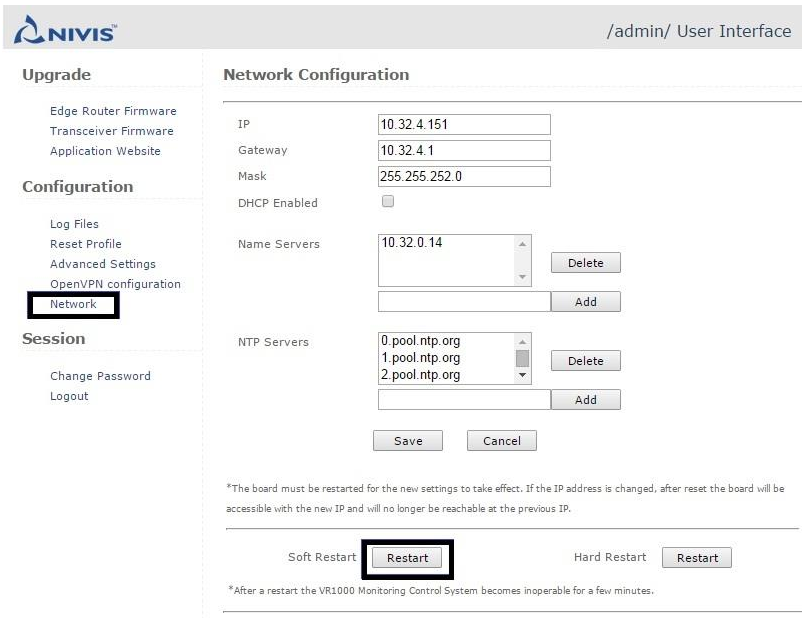
7.8.3.3 *Configure Quark modules to use open communication*

This step must be performed **after** setting the "transparent" mode, as setting the transparent mode automatically configures the Quark modules to use OpenVPN. After setting the "transparent" mode, the Quark software restarts rendering the web interface inaccessible for few tens of seconds. Wait until the web interface is accessible.

8. Point a web browser to the Quark /admin/ website (http://<quark_ip>/admin/), enter credentials
9. Go to "Advanced settings" page
10. Choose "BHManager" from "Section" drop-down
11. Choose "SECURITY" in "Variable" drop-down
12. Set the field "Value" to "none" (no quotes, not other char)



13. Go to "Network" page
14. Perform a "Soft Restart" to activate the changes


NIVIS
/admin/ User Interface
Upgrade
Edge Router Firmware
Transceiver Firmware
Application Website
Configuration
Log Files
Reset Profile
Advanced Settings
OpenVPN configuration
Network
Session
Change Password
Logout
Network Configuration
IP
10.32.4.151
Gateway
10.32.4.1
Mask
255.255.252.0
DHCP Enabled
Name Servers
10.32.0.14
Delete
Add
NTP Servers
0.pool.ntp.org
1.pool.ntp.org
2.pool.ntp.org
Delete
Add
Save
Cancel
*The board must be restarted for the new settings to take effect. If the IP address is changed, after reset the board will be accessible with the new IP and will no longer be reachable at the previous IP.
Soft Restart
Restart
Hard Restart
Restart
*After a restart the VR1000 Monitoring Control System becomes inoperable for a few minutes.

8 Use Cases

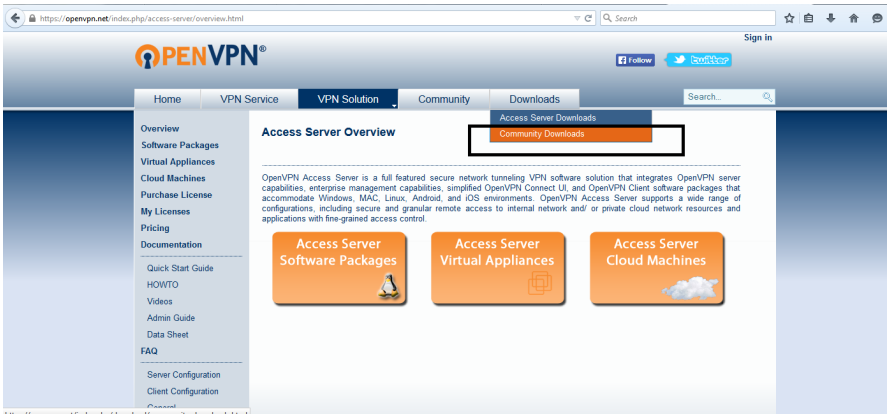
8.1 Setting the PC-to-device communication

The PC-to-Device communication is taking place via the following path: PC – NMS – Edge Router – Device. There are distinct settings to be made in case the PC runs Windows or Linux OS. The PC / NMS communication can be VPN-based or non-VPN based. The NMS / Edge Router are always VPN-based on real deployments, but for lab tests, it can be non-VPN (Setting up the NMS/VR1000 communication was covered above). The sections below describe possible combinations to set-up the remaining segment: PC/NMS.

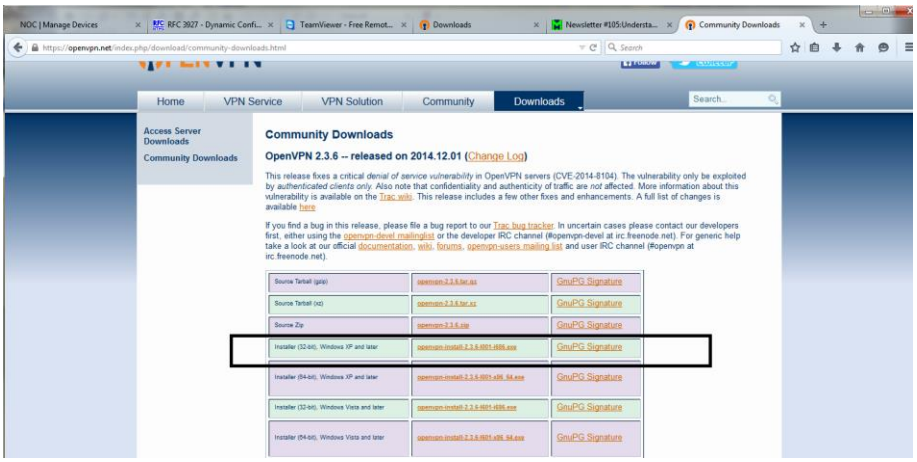
8.1.1 PC-to-device IPv6 communication for Windows user, VPN based

8.1.1.1 Install Open VPN on Windows

1. Go to https://openvpn.net/ , select 'VPN Solution' option and choose 'Community Downloads' tab option:



2. Download the release '*.exe' that suits user computer. For example, there was chosen the following openvpn for 32-bit windows computer:



3. Install it **as administrator** in the (preferably) default directory “C:\Program Files\OpenVPN”.

8.1.1.2 Setup Windows/NMS IPv4 communication VPN-based

1. Get the following certificates: **nms_ca.crt**, **nms_remote_user.crt**, **nms_remote_user.key** from the NMS administrator.
2. Create a configuration file ‘*.ovpn’ in the “c:\Program Files\OpenVPN\config” with the following information (replace placeholders “<**server_public_IPv4**>” and “<**server_port**>” with the appropriate values):

client

dev tun1500

proto tcp

remote <**server_public_IPv4**> <**server_port**>

resolv-retry infinite

nobind

ca nms_ca.crt

cert nms_remote_user.crt

key nms_remote_user.key

comp-lzo

ns-cert-type server

verb 3

3. Start **as administrator** the OpenVPN by executing the exe file: "C:\Program Files\OpenVPN\bin\openvpn-gui.exe"
4. After it was started, right click on the icon of the VPN application and choose to connect to VPN Server.
5. Test the VPN connection:
    a. Check if the interface with the assigned VPN ipv4 address <**client_VPN_IPv4**> was created:
        i. ipconfig
        ii. Check in the command output shows and IPv4 in the range **10.185.0.0/24**
    b. Check the VPN connection with server:
        i. ping **10.185.0.1**

NOTE
Use 1194 for <**server_port**> variable

The NMS server is configured by default to provide to the clients the <**client_VPN_IPv4**> in the range **10.185.0.0/24**

The NMS server is configured with <**server_VPN_IPv4**>: **10.185.0.1**

8.1.1.3 Setup Windows/NMS IPv6 communication VPN-based

1. Prepare to connect to NMS from Windows PC:
    a. Use **ipconfig** command to get the local VPN IPv4 address: <**client_VPN_IPv4**>, for example 10.185.0.14
    b. The server (NMS) VPN IPv4 address <**server_VPN_IPv4**> is known, configured by default as: **10.185.0.1**
    c. Compute the IPv6 address <**client_VPN_IPv6**> based on client VPN IPv4 address, using FD00 as prefix, and <**client_VPN_IPv4**> converted to hex as postfix. For example:
        i. Computed IPv6: FD00::**0AB9:000E**
            1. FD00 is a MUST
            2. **0AB9:000E** is **10.185.0.14** converted to hex
2. Connect to NMS:
    a. On **Windows PC** (on administrator console): create 6in4 tunnel
        i. netsh interface ipv6 add v6v4tunnel remote_user_tunn <**client_VPN_IPv4**> <**server_VPN_IPv4**>
        ii. netsh interface ipv6 add address remote_user_tunn <**client_VPN_IPv6**>/128
        iii. netsh interface ipv6 add route ::/0 remote_user_tunn
    b. O**n NMS:** create route to remote user machine
        i. sudo ip route add <**client_VPN_IPv6**>/128 via ::<**client_VPN_IPv4**>

Example using the variables chosen above:

    a. On **Windows PC** (on administrator console): create 6in4 tunnel
        i. netsh interface ipv6 add v6v4tunnel remote_user_tunn 10.185.0.14 10.185.0.1
        ii. netsh interface ipv6 add address remote_user_tunn FD00::0AB9:000E/128
        iii. netsh interface ipv6 add route ::/0 remote_user_tunn
    a. O**n NMS:** create route to remote user machine
        i. sudo ip route add FD00::0AB9:000E/128 via ::10.185.0.14

NOTE
All configurations are preserved if the OpenVPN Client is restarted

8.1.1.4 Test Windows PC/NMS IPv6 communication

1. Select an address of an already connected Edge Router or Smart Object as destination (ex. 2012:0:0:2::00fa:0162):
    a. **ping -6 2012:0:0:2::00fa:0162**

8.1.1.5 Cleaning up the remote access to NMS from Windows PC

In order to clean up the settings after the connectivity test was done, the following steps are necessary:

1. Disconnect from NMS:
    a. On **Windows PC** (on administrator console), close 6in4 tunnel:
        i. netsh interface ipv6 delete address remote_user_tunn <**client_VPN_IPv6**>/128
        ii. netsh interface ipv6 delete route ::/0 remote_user_tunn
        iii. netsh interface ipv6 delete remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        i. sudo ip route del <**client_VPN_IPv6**>/128 via ::<**client_VPN_IPv4**>

    Example using the variables chosen above:

    a. On **Windows PC** (on administrator console), close 6in4 tunnel:
        ii. netsh interface ipv6 delete address remote_user_tunn FD00::0AB9:000E/128
        iii. netsh interface ipv6 delete route ::/0 remote_user_tunn
        iv. netsh interface ipv6 delete remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        v. sudo ip route del FD00::0AB9:000E/128 via ::10.185.0.14

2. Remove VPN certificates and key for remote user from path c:\Program Files\OpenVPN\config on Windows PC.

8.1.2 PC-to-device IPv6 communication for Windows user, non-VPN based

The non-VPN based connectivity will only work if the PC can communicate directly via IPv4 with the NMS (the NMS IPv4 is visible to the PC)

8.1.2.1 Setup Windows/NMS IPv6 communication non-VPN-based

1. Test IPv4 connection with NMS
    a. On **NMS**, use **ifconfig** to get the remote(NMS) IPv4 address (<**server_IPv4**>), for example 10.32.4.233
    b. On **Windows PC**, test the connectivity with NMS:
        i. ping <**server_IPv4**>
2. Prepare to connect to NMS from remote user machine:
    a. On PC client, use **ipconfig** command to get the local IPv4 address: <**client_IPv4**>, for example 10.32.4.131
    b. Compute the IPv6 address <**client_IPv6**> based on client IPv4 address, using FD00 as prefix, and <**client_IPv4**> converted to hex as postfix. For example:
        i. Computed IPv6: FD00::**0A20:0483**
            1. FD00 is a MUST
            2. **0A20:0483** is **10.32.4.131** converted to hex
3. Connect to NMS:
    a. On **Windows PC** (on administrator console), create 6in4 tunnel:
        i. netsh interface ipv6 add v6v4tunnel remote_user_tunn <**client_IPv4**> <**server_IPv4**>
        ii. netsh interface ipv6 add address remote_user_tunn <**client_IPv6**>/128
        iii. netsh interface ipv6 add route ::/0 remote_user_tunn
    b. On **NMS** add route to Windows PC
        i. sudo ip route add <**client_IPv6**>/128 via ::<**client_IPv4**>

Example using the variables chosen above:

    a. On **Windows PC** (on administrator console), create 6in4 tunnel:
        i. netsh interface ipv6 add v6v4tunnel remote_user_tunn 10.32.4.131 10.32.4.233
        ii. netsh interface ipv6 add address remote_user_tunn FD00::0A20:0483/128
        iii. netsh interface ipv6 add route ::/0 remote_user_tunn
    b. On **NMS** add route to Windows PC
        i. sudo ip route add FD00::0A20:0483/128 via ::10.32.4.131

8.1.2.2 Test Windows PC/NMS IPv6 communication

1. Select an address of an already connected Edge Router or Smart Object as destination (ex. 2012:0:0:2::00fa:0162):

a. **ping -6 2012:0:0:2::00fa:0162**

8.1.2.3 Cleaning up the remote access to NMS from Windows PC

In order to clean up the settings after the connectivity test was done, the following steps are necessary:

1. Disconnect from NMS:
    a. On **Windows PC** (on administrator console), close 6in4 tunnel:
        i. netsh interface ipv6 delete address remote_user_tunn <**client_IPv6**>/128
        ii. netsh interface ipv6 delete route ::/0 remote_user_tunn
        iii. netsh interface ipv6 delete remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        i. sudo ip route del <**client_IPv6**>/128 via ::<**client_IPv4**>

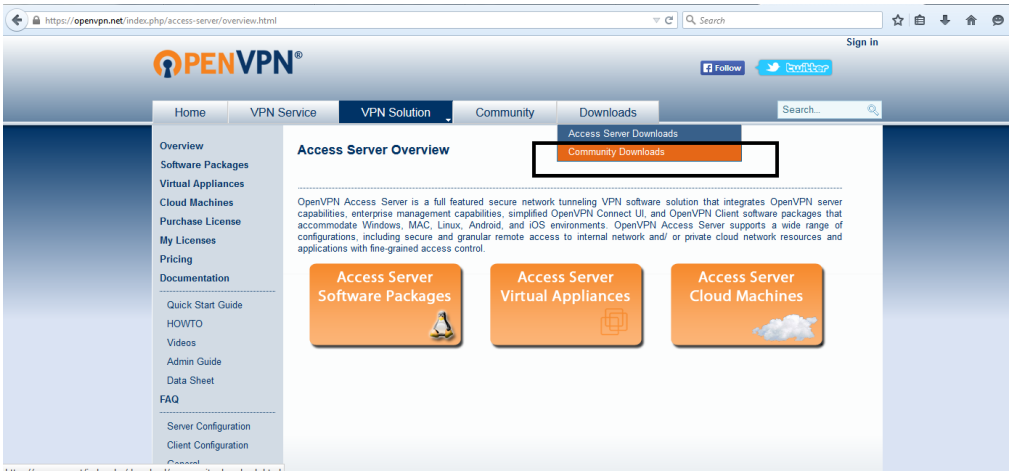
Example using the variables chosen above:

    c. On **Windows PC** (on administrator console), close 6in4 tunnel:
        ii. netsh interface ipv6 delete address remote_user_tunn FD00::0A20:0483/128
        iii. netsh interface ipv6 delete route ::/0 remote_user_tunn
        iv. netsh interface ipv6 delete remote_user_tunn
    d. On **NMS,** remove route to Windows PC:
        i. sudo ip route del FD00::0A20:0483/128 via :: 10.32.4.131

8.1.3 PC-to-device IPv6 communication for Linux user, VPN based

8.1.3.1 Install OpenVPN on Linux

1. Go to https://openvpn.net/ , select 'VPN Solution' option and choose 'Community Downloads' tab option:



2. Download the release '*.tar.gz' that suits user computer. For example, openvpn for 32-bit Linux computer:

3. Compile and install it as follows:
   a. tar xfz openvpn-[version].tar.gz
   b. cd 'top_level_directory'
   c. ./configure
   d. make

e. make install

8.1.3.2 Setup Linux/NMS IPv4 communication VPN-based

1. Get the following certificates: **nms_ca.crt**, **nms_remote_user.crt**, **nms_remote_user.key** from the NMS administrator.
2. Create a configuration file '*.conf' with the following information (replace placeholders "<**server_public_IPv4**>" and "<**server_port**>" with the appropriate values):

client

dev tun1500

proto tcp

remote <**server_public_IPv4**> <**server_port**>

resolv-retry infinite

nobind

ca nms_ca.crt

cert nms_remote_user.crt

key nms_remote_user.key

comp-lzo

ns-cert-type server

verb 3

3. Start **as administrator** the OpenVPN
    a. sudo openvpn [path_to_config_file]
4. Test the VPN connection:
    a. Check if the interface with the assigned VPN ipv4 address <**client_VPN_IPv4**> was created:
        i. ipconfig
        ii. check in the command output shows and IPv4 in the range **10.185.0.0/24**
    b. Check the VPN connection with server:
        i. ping **10.185.0.1**

NOTE

Use 1194 for <**server_port**> variable

The NMS server is configured by default to provide to the clients the <**client_VPN_IPv4**> in the range **10.185.0.0/24**

The NMS server is configured with <**server_VPN_IPv4**>: **10.185.0.1**

8.1.3.3 Setup Linux/NMS IPv6 communication VPN-based

1. Prepare to connect to NMS from Windows PC:
    a. Use **ipconfig** command to get the local VPN IPv4 address: <**client_VPN_IPv4**>, for example 10.185.0.38
    b. The server (NMS) VPN IPv4 address <**server_VPN_IPv4**> is known, configured by default as: **10.185.0.1**
    c. Compute the IPv6 address <**client_VPN_IPv6**> based on client VPN IPv4 address, using FD00 as prefix, and <**client_VPN_IPv4**> converted to hex as postfix. For example:
        i. Computed IPv6: FD00::**0AB9:0026**
            1. FD00 is a MUST
            2. **0AB9:000E** is **10.185.0.38** converted to hex
2. Connect to NMS:
    a. On **client Linux PC**: create 6in4 tunnel
        i. sudo ip tunnel add remote_user_tunn mode sit local <**client_VPN_IPv4**> remote <**server_VPN_IPv4**>
        ii. sudo ip addr add <**client_VPN_IPv6**>/128 dev remote_user_tunn
        iii. sudo ip route add ::/0 dev remote_user_tunn
        iv. sudo ip link set dev remote_user_tunn up
    b. O**n NMS:** create route to remote user machine
        i. sudo ip route add <**client_VPN_IPv6**>/128 via ::<**client_VPN_IPv4**>

Example using the variables chosen above:

    b. On **client Linux PC** (on administrator console): create 6in4 tunnel
        i. sudo ip tunnel add remote_user_tunn mode sit local 10.185.0.38 remote 10.185.0.1
        ii. sudo ip addr add FD00::0AB9:0026/128 dev remote_user_tunn
        iii. sudo ip route add ::/0 dev remote_user_tunn
        iv. sudo ip link set dev remote_user_tunn up
    a. O**n NMS:** create route to remote user machine
        i. sudo ip route add FD00::0AB9:0026/128 via ::10.185.0.38

NOTE
All configurations are preserved if the OpenVPN Client is restarted

8.1.3.4 Test Linux PC/NMS IPv6 communication

1. Select an address of an already connected Edge Router or Smart Object as destination (ex. 2012:0:0:2::00fa:0162):

a. **ping6 2012:0:0:2::00fa:0162**

8.1.3.5 Cleaning up the remote access to NMS from Linux PC

In order to clean up the settings after the connectivity test was done, the following steps are necessary:

1. Disconnect from NMS:
    a. On **client Linux PC**, close 6in4 tunnel:
        i. sudo ip addr del <**client_VPN_IPv6**>/128 dev remote_user_tunn
        ii. sudo ip route del ::/0 dev remote_user_tunn
        iii. sudo ip tunnel del remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        i. sudo ip route del <**client_VPN_IPv6**>/128 via ::<**client_VPN_IPv4**>

    Example using the variables chosen above:

    a. On **client Linux PC**, close 6in4 tunnel:
        i. sudo ip addr del FD00::0AB9:0026/128 dev remote_user_tunn
        ii. sudo ip route del ::/0 dev remote_user_tunn
        iii. sudo ip tunnel del remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        i. sudo ip route del FD00::0AB9:0026/128 via ::10.185.0.38

2. Remove VPN certificates and key from the client Linux PC.

8.1.4 PC-to-device IPv6 communication for Linux user, non-VPN based

The non-VPN based connectivity will only work if the PC can communicate directly via IPv4 with the NMS (the NMS IPv4 is visible to the PC)

8.1.4.1 Setup Linux/NMS IPv6 communication non-VPN-based

1. Test IPv4 connection with NMS
    a. On **NMS**, use **ifconfig** to get the remote(NMS) IPv4 address (<**server_IPv4**>), for example 10.32.4.233
    b. On **Windows PC**, test the connectivity with NMS:
        i. ping <**server_IPv4**>
2. Prepare to connect to NMS from remote user machine:
    a. On PC client, use **ipconfig** command to get the local IPv4 address: <**client_IPv4**>, for example 10.32.4.17
    b. Compute the IPv6 address <**client_IPv6**> based on client IPv4 address, using FD00 as prefix, and <**client_IPv4**> converted to hex as postfix. For example:
        i. Computed IPv6: FD00::**0A20:0411**
            1. FD00 is a MUST
            2. **0A20:0483** is **10.32.4.17** converted to hex
3. Connect to NMS:
    a. On **client Linux PC**: create 6in4 tunnel
        i. sudo ip tunnel add remote_user_tunn mode sit local <**client_IPv4**> remote <**server_IPv4**>
        ii. sudo ip addr add <**client_IPv6**>/128 dev remote_user_tunn
        iii. sudo ip route add ::/0 dev remote_user_tunn
        iv. sudo ip link set dev remote_user_tunn up
    b. O**n NMS:** create route to remote user machine
        i. sudo ip route add <**client_ IPv6**>/128 via ::<**client_ IPv4**>

Example using the variables chosen above:

    a. On **client Linux PC** (on administrator console): create 6in4 tunnel
        i. sudo ip tunnel add remote_user_tunn mode sit local 10.185.0.17 remote 10.185.0.1
        ii. sudo ip addr add FD00::0A20:0411/128 dev remote_user_tunn
        iii. sudo ip route add ::/0 dev remote_user_tunn
        iv. sudo ip link set dev remote_user_tunn up
    b. O**n NMS:** create route to remote user machine
        i. sudo ip route add FD00::0A20:0411/128 via ::10.185.0.17

8.1.4.2 Test Windows PC/NMS IPv6 communication

1. Select an address of an already connected Edge Router or Smart Object as destination (ex. 2012:0:0:2::00fa:0162):
    a. **ping6 2012:0:0:2::00fa:0162**

8.1.4.3 Cleaning up the remote access to NMS from Linux PC

1. Disconnect from NMS:
    a. On **client Linux PC**, close 6in4 tunnel:
        i. sudo ip addr del <**client_VPN_IPv6**>/128 dev remote_user_tunn
        ii. sudo ip route del ::/0 dev remote_user_tunn
        iii. sudo ip tunnel del remote_user_tunn
    b. On **NMS,** remove route to Windows PC:
        i. sudo ip route del <**client_VPN_IPv6**>/128 via ::<**client_VPN_IPv4**>

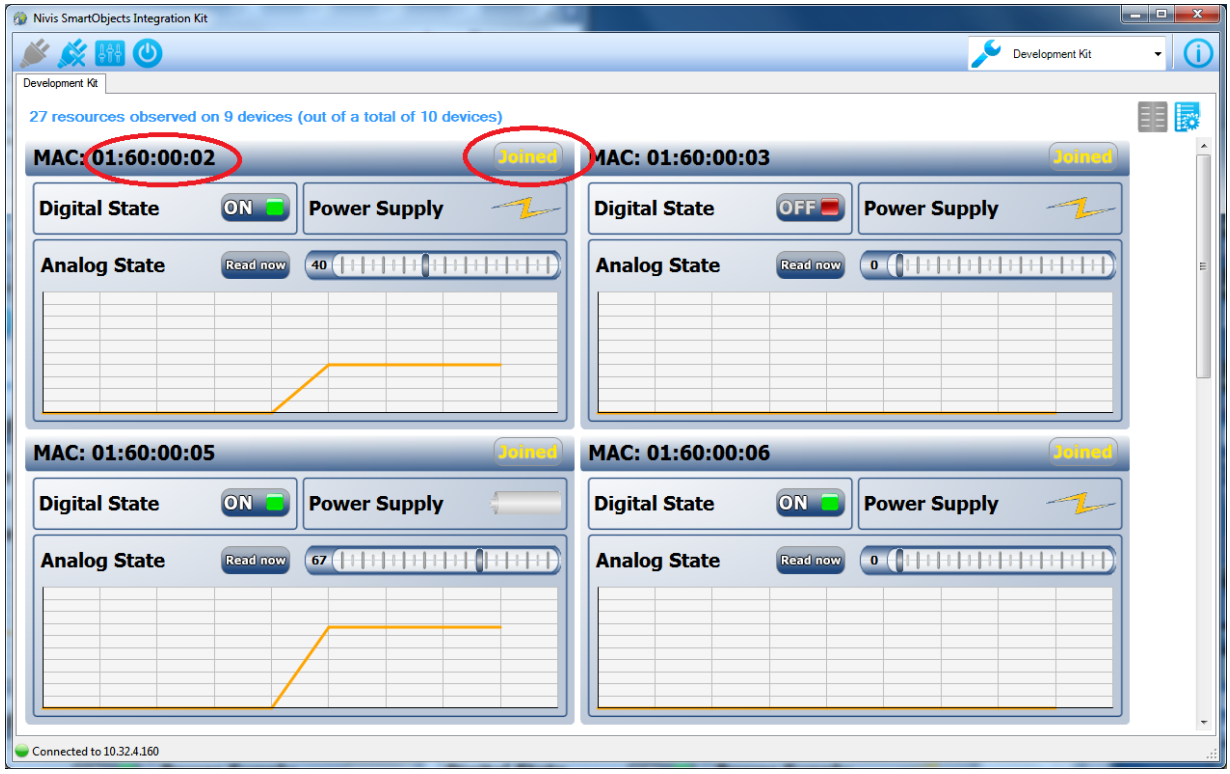
Example using the variables chosen above:

    c. On **client Linux PC**, close 6in4 tunnel:
        iv. sudo ip addr del FD00::0A20:0411/128 dev remote_user_tunn
        v. sudo ip route del ::/0 dev remote_user_tunn
        vi. sudo ip tunnel del remote_user_tunn
    d. On **NMS,** remove route to Windows PC:
        ii. sudo ip route del FD00::0A20:0411/128 via ::10.32.4.17

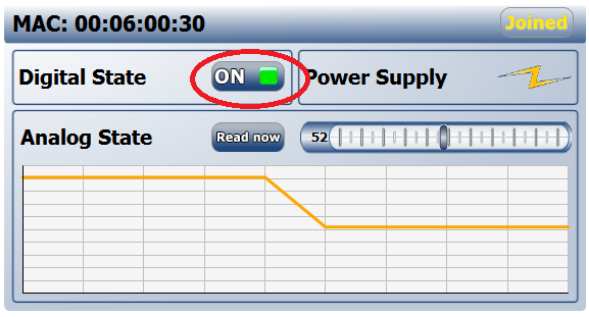
8.2 Evaluating the on-board application COAP mapped parameters

Once the Smart Object network is up and running, a first evaluation scenario consists of simply observing changes in the three pre-defined application resources.

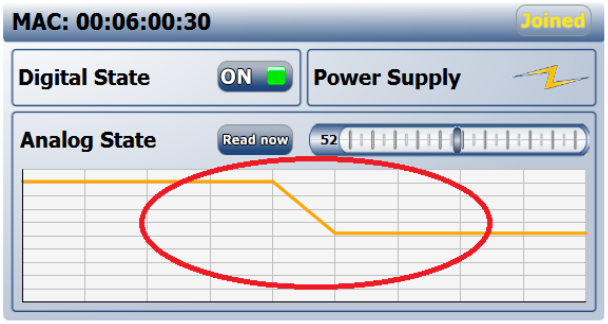
1. Ensure in the NAMT that the SOEN is joined to the network (its MAC appears in the Development Kit, with status as Joined and yellow font). An example for device with MAC ending with "01:06:00:02" is shown below.



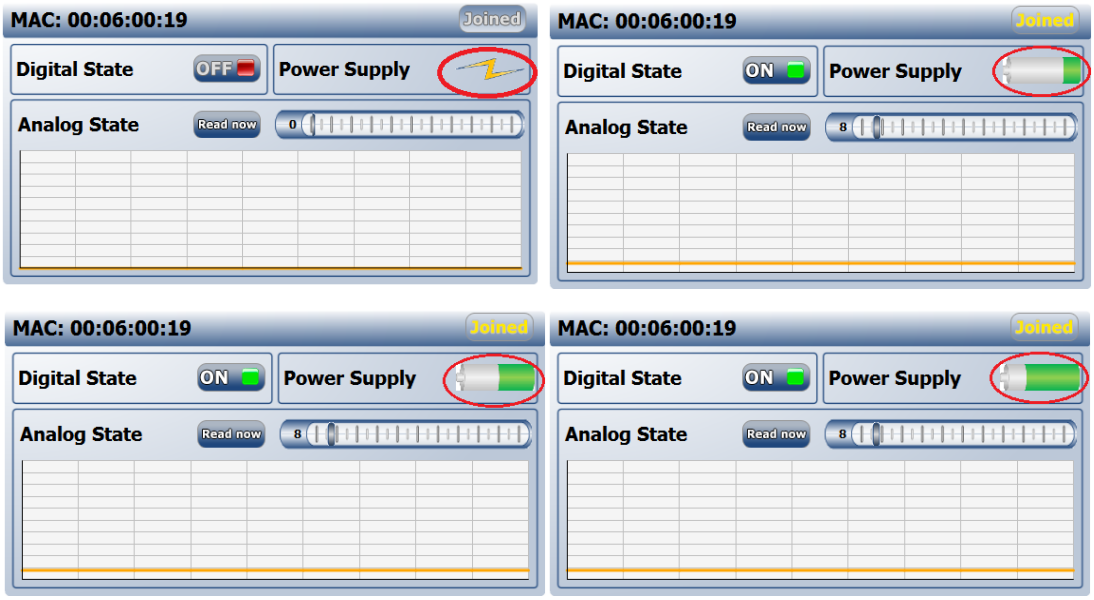
2. Toggle the digital ON/OFF switch (SW2) present on the power pack board every 5 seconds. Observe the change in value in the NAMT Development Kit pane.



3. Vary the potentiometer R21 present on the power pack board by turning it by more than 45 degrees. Observe the change in value in the NAMT Development Kit pane.



4. The line-powered or battery-powered status of the Smart Object is shown in the **Power supply** area. For battery-powered Smart Objects, the battery level estimate is shown.



8.3 Application Processor Mapped COAP parameters

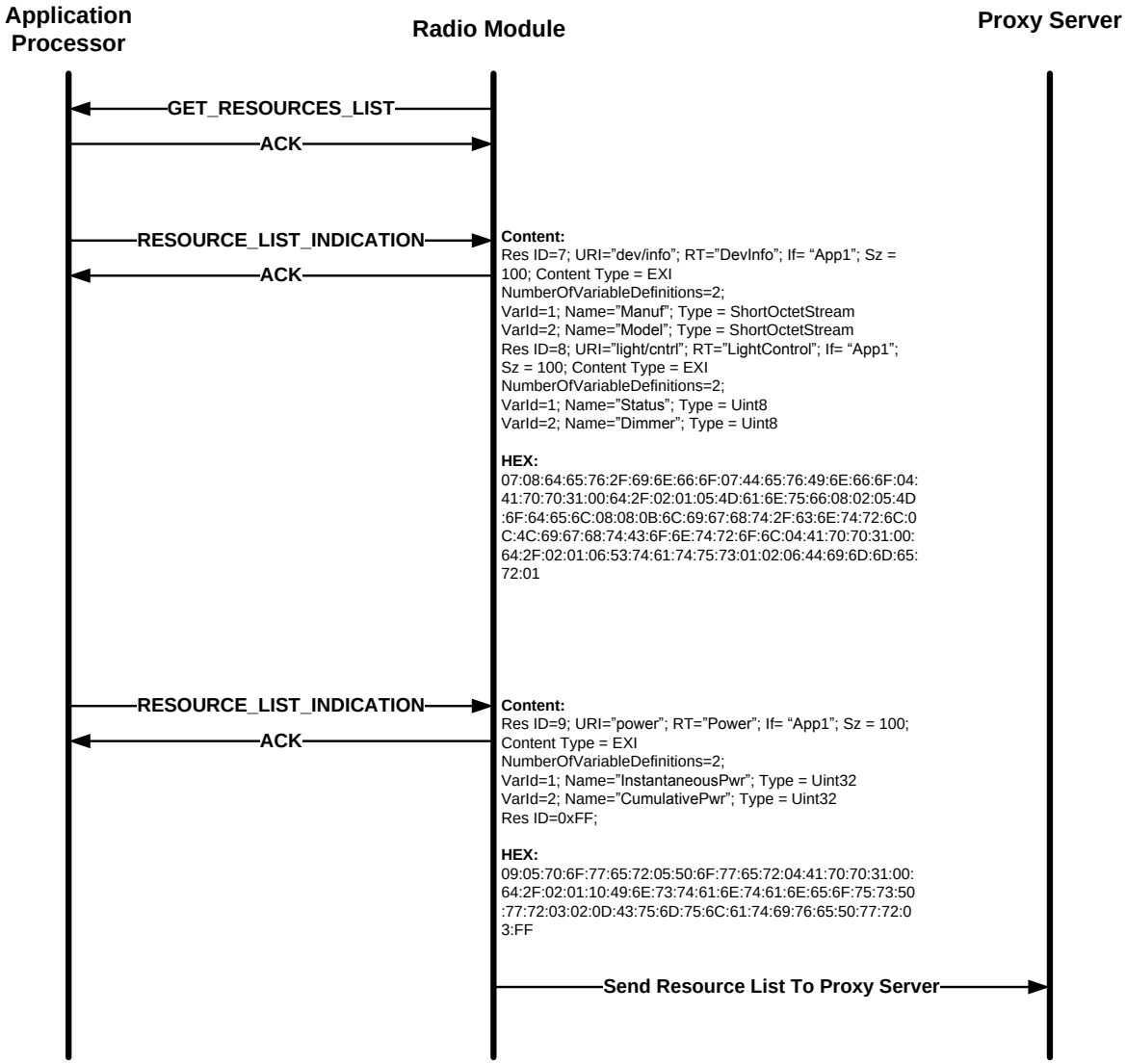
8.3.1 Overview

The Application Processor can map process variables as COAP resources for the external application to read or observe.

The flow of packets through the network during resource discovery, read resource, and other processes is described below.

8.3.2 Resource Discovery

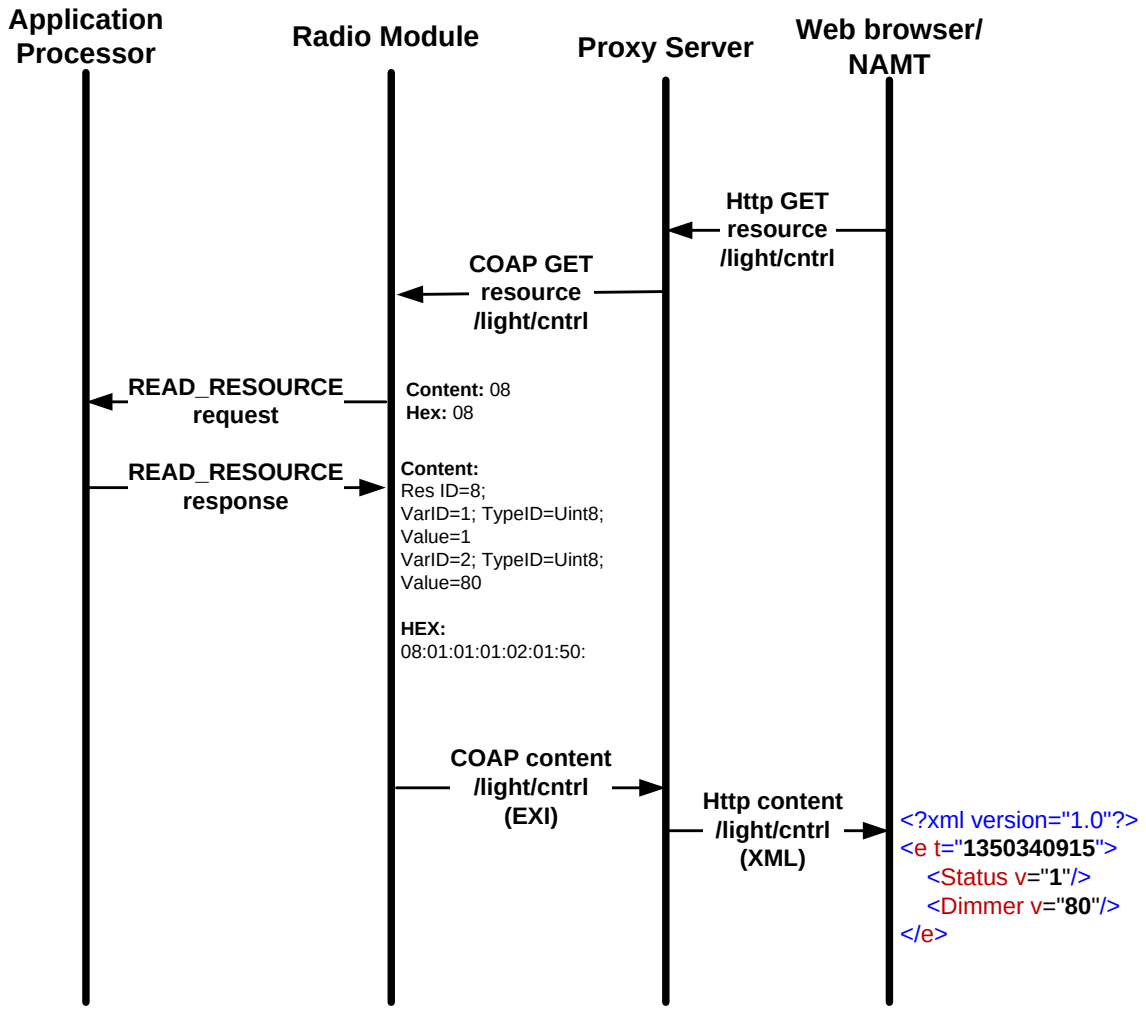
This section describes the sequence of messages exchanged between the Radio Module (RM) and the Application Processor (AP) from device start until it uploads the list of resources to the proxy server.



1. After power ON, the Radio Module CPU sends a GET_RESOURCE_LIST command to the Application Processor. It will keep sending this command until it receives ACK/NACK from the AP.
2. After sending the response to the previous command (either in the form of an ACK or in the form of a NACK), the AP starts sending RESOURCE_LIST_INDICATION commands to the RM. Each of these commands will be acknowledged by the RM.
3. The RM receives the list of resources defined by the AP, terminated by a resource with ID 0xFF. After the RM receives resource ID 0xFF, any following resources will be rejected with a NACK message.

8.3.3 Read Resource

This section describes the sequence of messages exchanged when the application processor has to respond to a READ_RESOURCE request command.

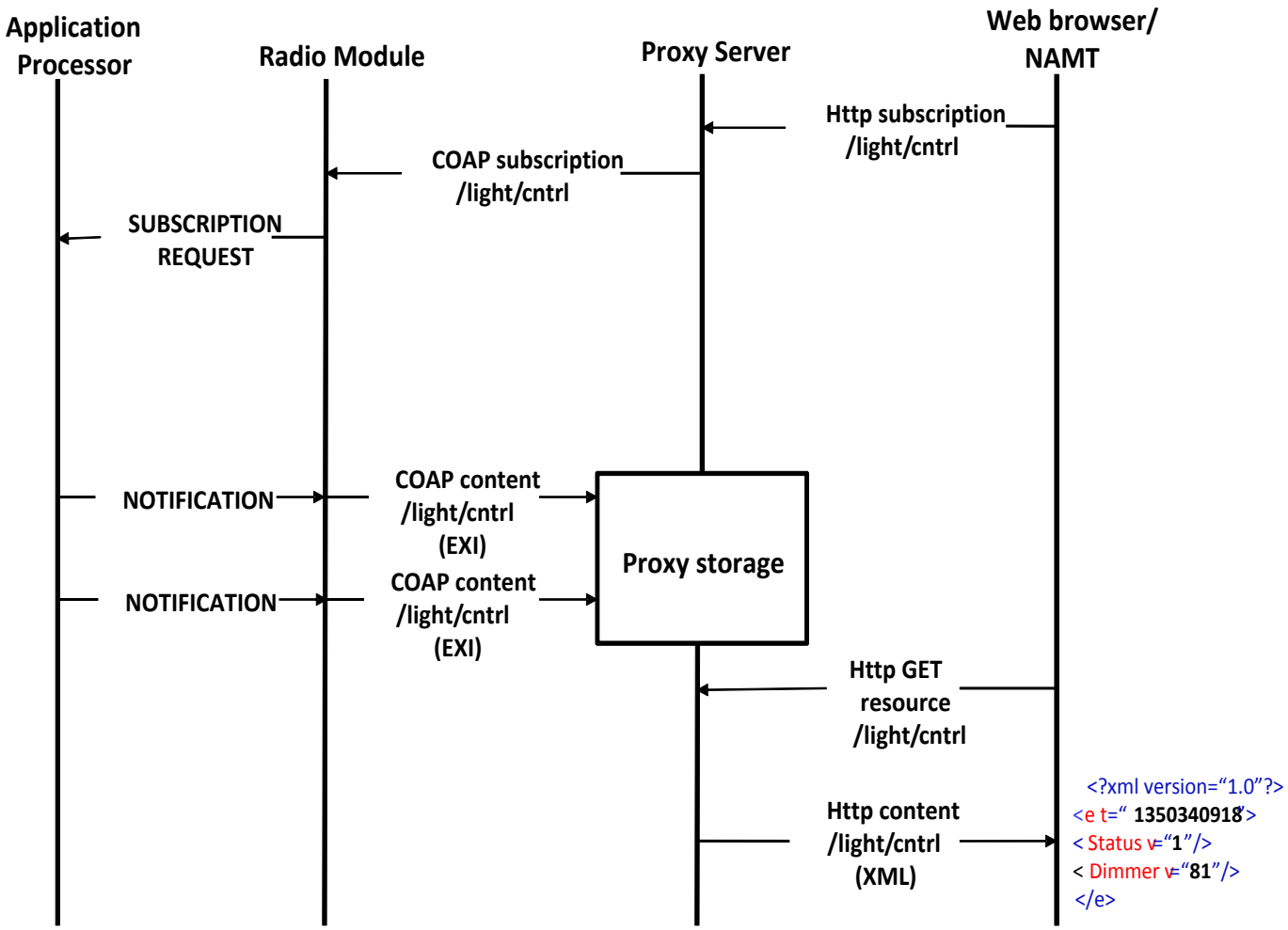


1. The Radio Modem sends a READ_RESOURCE request to the Application Processor, specifying the resource ID.

2. The Application Processor responds with the resource representation at that moment in time.

8.3.4 Observe Resource

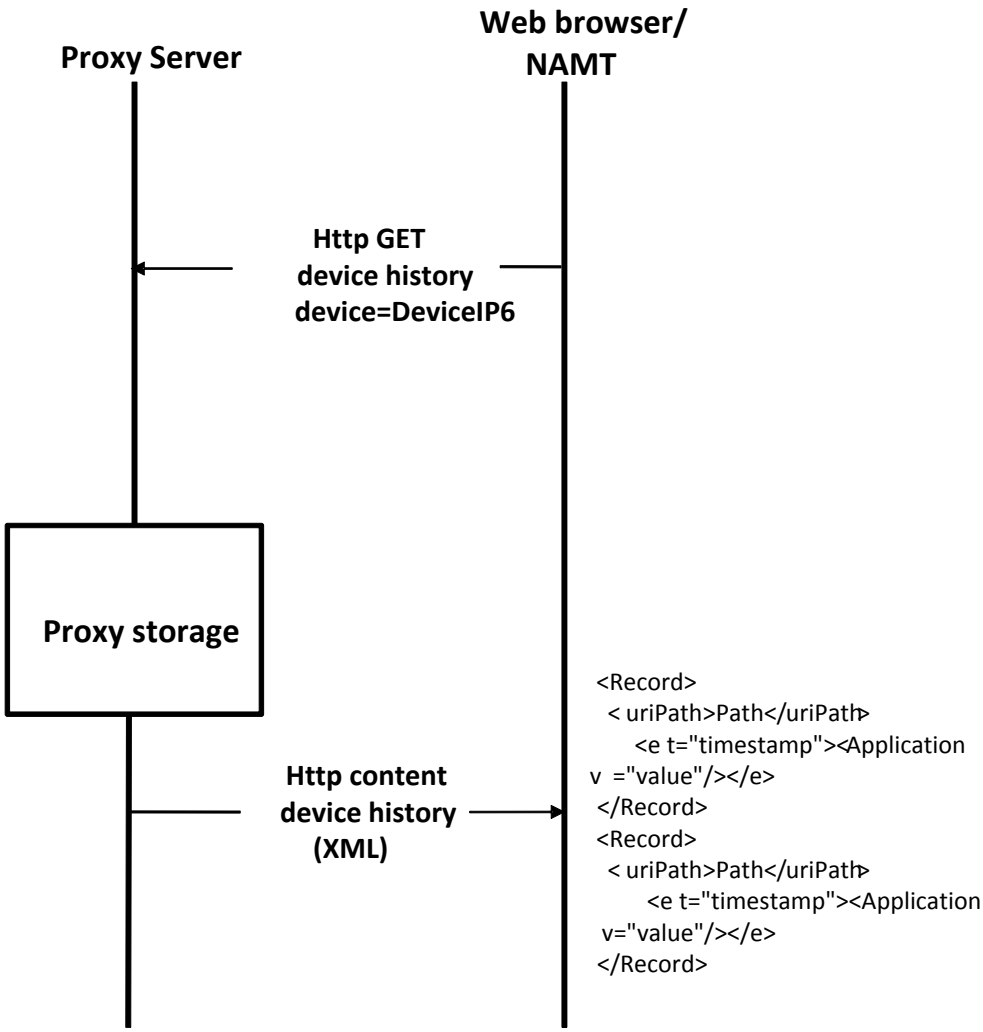
This section describes the sequence of messages exchanged when the application processor has to respond to a SUBSCRIPTION_REQUEST.



1. The Radio Modem sends a SUBSCRIPTION_ REQUEST to the Application Processor, specifying the resource ID.
2. The Application Processor publishes the resource representation at every publish interval or when resource is modified.

8.3.5 Retrieve Device History

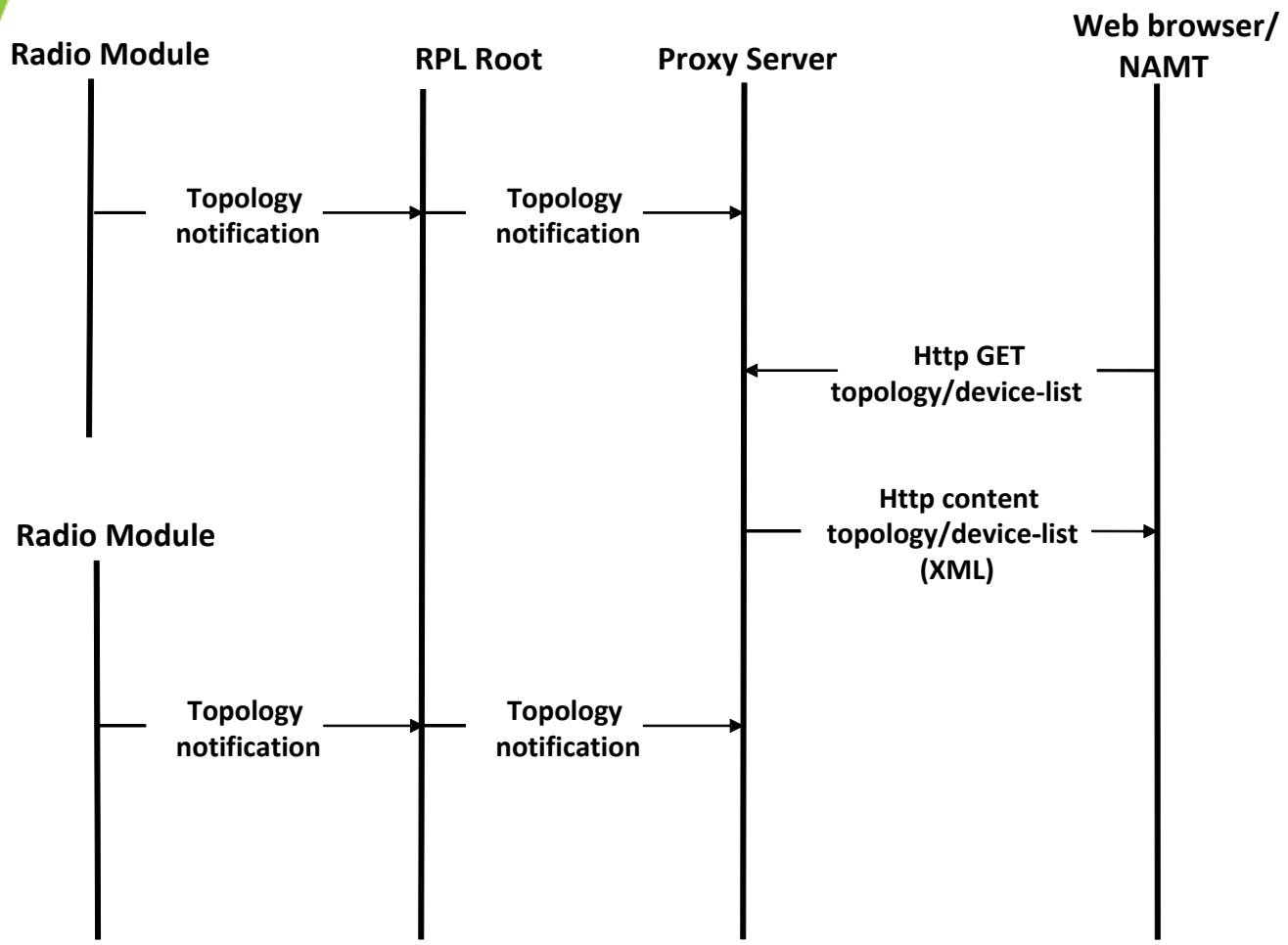
This section describes the sequence of messages exchanged when the Proxy Server receives a request for retrieving data history.



1. The web browser or NAMT sends an Http GET device history request to the Proxy Server, specifying the device IPv6 address.
2. The Proxy Server responds with an XML, representing the data history for that device.

8.3.6 Get network topology/device-list

This section describes the way Proxy Server constructs network topology.



1. Every time a radio device joins the network, it sends some topology notifications to RPL Root.
2. RPL Root forwards the topology notifications to the Proxy Server.
3. The Proxy Server responds to topology/device-list requests from the web browser/NAMT.

9 Troubleshooting

The following table represents some typical problems that may be encountered while working with the Smart Object Development Kit, the common cause of each problem, and some possible solutions.

Problem	Explanation	Solution
No device joins the network (not even Transceiver)	The Transceiver module present on the Versa Router™ 1000 Quark is not connected properly to the antenna.	Ensure that the Transceiver is properly connected to the antenna.
No SO device joins the network, but Transceiver does join the network	The Net ID on the Smart Objects does not match the Vendor ID on the Quark.	Ensure consistency between SO Network ID and Quark Vendor ID (consistent with default settings).
Network slow to form, unstable after join	Smart Object or Transceiver on Quark has improperly connected antenna.	Make sure the antenna is properly connected.
With multiple coexisting networks, network slow to form, unstable after join	Overlapping networks with the same NetworkID/PanID	The NetworkID/PanID depends on the last byte of Versa Router™ 1000 Quark IPv6.
Network unstable after connecting NAMT in Development Kit mode	Too aggressive interrogation rate	Reduce the number of devices or increase the interrogation interval (Settings→Development Kit Refresh interval).
NAMT in Development Kit mode slow to update after powering off a Smart Object	There is no unjoin notification; NAMT keeps requesting COAP resources from devices powered off.	Restart the Versa Router™ 1000 Quark to rebuild the resource list.
Transceiver FW upgrade from NAMT fails at start, immediately after setting the time from NAMT	Did not wait a few minutes after setting the time from NAMT before connecting to the kit again.	Do not start the Transceiver firmware upgrade immediately after the Transceiver appears in the Network view; wait half a minute.
Inconsistent COAP resources shown after a Quark firmware upgrade	Smart Objects were not power cycled after Quark FW upgrade.	Restart all Smart Objects after Quark FW upgrade.
NAMT refuse to connect to Quark after setting time on Quark	The PC time changed significantly after setting the time on the Quark, resulting in time differences between PC and Quark, which lead to undefined network behavior.	Power cycle the whole system (Quark and Smart Objects).

Appendix A: List of Standards Supported in the Smart Object Platform

Standardization Body	Standard Designator	Revision	Title
IEEE	802.15.4g-2012	Final	Low-Rate Wireless Personal Area Networks (WPANs) Amendment 4: Physical Layer Specifications for Low Data Rate Wireless Smart Metering Utility Networks
IEEE	802.15.4e-2012	Final	Wireless Medium Access Control(MAC) and Physical Layer (PHY)Specifications for Low-Rate Wireless Personal Area Networks (WPANs)
IETF	RFC 2460	Final	Internet Protocol, Version 6 (IPv6)Specification
IETF	RFC 4443	Final	Internet Control Message Protocol for the IPv6 Specification
IETF	RFC 4944	Final	Transmission of IPv6 Packets over IEEE 802.15.4 Networks
IETF	RFC 6282	Final	Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks
IETF	RFC 6775	Final	Neighbor Discovery Optimization for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)
IETF	RFC 6550	Final	RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks
IETF	RFC 6202	Final	The Trickle Algorithm
IETF	RFC 6552	Final	Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)
IETF	RFC 6551	Final	Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks
IETF	RFC768	Final	User Datagram Protocol (UDP)
IETF	ID.draft-ietf-core-coap	Rev 16	Constrained Application Protocol (CoAP)
IETF	RFC 6690	Final	Constrained RESTful Environments (CoRE) Link Format
IETF	ID.draft-ietf-core-observe	Rev 8	Observing Resources in CoAP
W3C	W3C Recommendation 10 March 2011	1.0	Efficient XML Interchange (EXI)
IETF	RFC 6347	Final	Datagram Transport Layer Security Version 1.2
IETF	RFC 5191	Final	Protocol for Carrying Authentication for Network Access (PANA)
IETF	RFC 3748	Final	Extensible Authentication Protocol (EAP)
IETF	RFC 5216	Final	EAP-TLS Authentication Protocol
IETF	RFC 5246	Final	Transport Layer Security (TLS) Protocol Version 1.2

Appendix B: FCC Related Statements

General Statements (For all devices):

Warning: Changes or modifications to this device not expressly approved by Nivis™ LLC could void the user's authority to operate the equipment.

FCC-Specific Statements:

For Class B Devices:

NOTE This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

RF Exposure

This equipment complies with FCC radiation exposure limits set forth for an uncontrolled environment. This equipment should be installed and operated with minimum distance 20cm between the radiator and your body. This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.

Appendix C: Industry Canada Related Statements

Industry Canada Specific Statements:

Antenna (General):

Under Industry Canada regulations, this radio transmitter may only operate using an antenna of a type and maximum (or lesser) gain approved for the transmitter by Industry Canada. To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that necessary for successful communication.

Conformément à la réglementation d'IndustrieCanada, le présent émetteur radio peut fonctionner avec une antenne d'un type et d'un gain maximal (ou inférieur) approuvé pour l'émetteur par Industrie Canada. Dans le but de réduire les risques de brouillage radioélectrique à l'intention des autres utilisateurs, il faut choisir le type d'antenne et son gain de sorte que la puissance isotrope rayonnée équivalente (p.i.r.e.) ne dépasse pas l'intensité nécessaire à l'établissement d'une communication satisfaisante.

Detachable Antenna:

This radio transmitter (identify the device by certification number, or model number if Category II) has been approved by Industry Canada to operate with the antenna types listed below with the maximum permissible gain and required antenna impedance for each antenna type indicated. Antenna types not included in this list, having a gain greater than the maximum gain indicated for that type, are strictly prohibited for use with this device.

Le présent émetteur radio (identifier le dispositifpar son numéro de certification ou son numéro de modèle s'il fait partie du matériel de catégorieI) a été approuvé par Industrie Canada pour fonctionner avec les types d'antenne énumérés ci-dessous et ayant un gain admissible maximal et l'impédance requise pour chaque type d'antenne. Les types d'antenne non inclus dans cette liste, ou dont le gain est supérieur au gain maximal indiqué, sont strictement interdits pour l'exploitation de l'émetteur.

Approved Antennas:

Type	Gain
½ Wave Whip	2 dBi

Compliance:

This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes : (1) l'appareil ne doit pas produire de brouillage, et (2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

Label Information

Additional Requirements for Modular Approved Devices:

When a modular device is used in an end product where the label of the module is not visible, the host device must have an exterior label to include the information shown below. If a label for the host device is not available, the manual must provide information to the integrator on labeling requirements.

Contains FCC ID: SQB-NIVISVN400

Contains IC: 6546A-NIVISVN400

Appendix D: European (ETSI) Related Statements

The Nivis™ LLC model VersaNode 400 operates on a frequency of 863.00 to 870.00 MHz at 25 mW maximum output power and is therefore classified as a short-range device, of which the parameters to comply with are described in ERC/DEC 70-03.

The relevant ETSI standards, applicable to this type of equipment, as indicated in Annex 1 of ERC/DEC 70-03, are:

- EN 300 220-1 V2.4.1 (2012-05): Electromagnetic compatibility and radio spectrum matters (ERM); Short range devices (SRD); Radio equipment to be used in the 25 MHz to 1000 MHz frequency range with power levels ranging up to 500 mW; Part 1: Technical characteristics and test methods.
- EN 300 220-2 V2.4.1 (2012-05): Electromagnetic compatibility and radio spectrum matters (ERM); Short range devices (SRD); Radio equipment to be used in the 25 MHz to 1000 MHz frequency range with power levels ranging up to 500 mW; Part 2: Harmonized EN covering essential requirements under article 3.2 of the R&TTE Directive.

Declaration of Conformity

EU Directive 2004/108/EC (EMC Directive)
EU Directive 2006/95/EC (LV Directive)

Manufacturer/Authorized Representative: **Nivis, LLC**
200 Galleria Parkway, Suite 440
Atlanta, GA 30339

We, **NIVIS, LLC** declare on our sole responsibility, that the following product

Product Name: VersaNode 400
Model Number(s): VersaNode 400

Is in compliance with the essential requirements of the EMC Directive (2004/108/EC) as outlined in Annex III, and the Low Voltage Directive (2006/95/EC).

The product identified above complies with the EU Directive 2004/108/EC by meeting the Protection requirements concerning EMC:

Applied Standards: **EN 300-220-1 V2.4.1 & EN 300-220-2 V2.4.1**

The product identified above complies with the EU Directive 2006/95/EC by meeting the Protection requirements concerning Product Safety:

Applied Standards: **Not Applicable**

Countries Covered: **EU**: Austria, Belgium, Bulgaria, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, The Netherlands, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, United Kingdom

EFTA: Switzerland, Iceland, Liechtenstein, and Norway

_____
Place and Date

_____
Signature of Manufacturer/Authorized Representative

_____
Printed Name: